DISSERTATION

CHARACTERIZING THE VISIBLE ADDRESS SPACE TO ENABLE EFFICIENT CONTINUOUS IP GEOLOCATION

Submitted by

Manaf Gharaibeh

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2020

Doctoral Committee:

Advisor: Christos Papadopolous
Co-Advisor: Craig Partridge

John Heidemann
Indrakshi Ray
Stephen Hayne

ABSTRACT

CHARACTERIZING THE VISIBLE ADDRESS SPACE TO ENABLE EFFICIENT CONTINUOUS IP GEOLOCATION

Internet Protocol (IP) geolocation is vital for location-dependent applications and many network research problems. The benefits to applications include enabling content customization, proximal server selection, and management of digital rights based on the location of users, to name a few. The benefits to networking research include providing geographic context useful for several purposes, such as to study the geographic deployment of Internet resources, bind cloud data to a location, and to study censorship and monitoring, among others.

The measurement-based IP geolocation is widely considered as the state-of-the-art client-independent approach to estimate the location of an IP address. However, full measurement-based geolocation is prohibitive when applied continuously to the entire Internet to maintain up-to-date IP-to-location mappings. Furthermore, many IP address blocks rarely move, making it unnecessary to perform such full geolocation.

The thesis of this dissertation states that *we can enable efficient, continuous IP geolocation by identifying clusters of co-located IP addresses and their location stability from latency observations*. In this statement, a cluster indicates a group of an arbitrary number of adjacent co-located IP addresses (a few up to a /16). Location stability indicates a measure of how often an IP block changes location. We gain efficiency by allowing IP geolocation systems to geolocate IP addresses as units, and by detecting when a geolocation update is required, optimizations not explored in prior work. We present several studies to support this thesis statement.

We first present a study to evaluate the reliability of router geolocation in popular geolocation services, complementing prior work that evaluates end-hosts geolocation in such services. The results show the limitations of these services and the need for better solutions, motivating our work to enable more accurate approaches. Second, we present a method to identify clus-

ters of *co-located* IP addresses by the similarity in their latency. Identifying such clusters allows us to geolocate them efficiently as units without compromising accuracy. Third, we present an efficient delay-based method to identify IP blocks that move over time, allowing us to recognize when geolocation updates are needed and avoid frequent geolocation of the entire Internet to maintain up-to-date geolocation. In our final study, we present a method to identify cellular blocks by their distinctive variation in latency compared to WiFi and wired blocks. Our method to identify cellular blocks allows a better interpretation of their latency estimates and to study their geographic properties without the need for proprietary data from operators or users.

ACKNOWLEDGEMENTS

I wish to show my sincere gratitude to the people who helped me on the path towards this dissertation.

I am particularly grateful to my advisor, Dr. Christos Papadopolous. Under his guidance, I have had the chance to explore exciting topics in network measurement and security and to work with talented researchers from other research groups. I am thankful for his mentorship, encouragement, kindness, insights into the specifics of my work, and emphasis on rigorous research. Going forward, I hope to continue having Christos as a mentor and to build on this fruitful relationship.

My sincere thanks to Dr. Craig Partridge, whom I have been honored to have as my co-advisor for the past two years. Craig's guidance and support proved monumental towards improving and finishing this dissertation, I will always be indebted to him for that. I am also indebted to Dr. John Heidemann, USC/ISI. John has always been willing to provide insight and direction in the past five years. He helped me become a better researcher. I am grateful for all of the feedback he provided to help me improve this document, and I appreciate his trip from the West Coast to attend my dissertation oral-defense. I also appreciate the valuable learning experience I have had from participating in the weekly meetings of his research group. I also thank John's group for making much of the measurement data I used in this dissertation available.

I thank, Christos, Craig, John, once more, for making me want to know so much more and for showing me the way to be a scientist.

I wish to thank Dr. Indrakshi Ray and Dr. Stephen Hayne for their service on my dissertation committee. I thank my committee members for the insights, feedback, and discussions that helped me improve this dissertation. Thanks are also due to all current and former professors at CSU who helped me along the way, including Dr. Charles Anderson, Dr. Sanjay Rajopadhye, Dr. Dan Massey, Dr. Joseph Gersch, Dr. Lorenzo De Carli, Dr. Indrajit Ray, and Dr. Robert France.

I am grateful to all my friends in the network-security research group for making the Ph.D. life a little easier, and for all the interesting discussions about research and other innumerable random topics. I extend my gratitude to all other friends in the Department of Computer Science, CSU, and the good old comrades anywhere for their support and encouragement.

Most importantly, I wish to acknowledge the endless support and encouragement of my family, who kept me going. I owe it all to my beloved parents, Hamed and Fawzeyah. The support of my sister, Manal, and my two brothers, Sameer and Ehab, has been invaluable, “Shokran”.

DEDICATION

to my parents ...

TABLE OF CONTENTS

ABSTRACT ii
ACKNOWLEDGEMENTS iv
DEDICATION vi
LIST OF TABLES x
LIST OF FIGURES xi

Chapter 1 Introduction 1
1.1 Problems Addressed 2
1.2 Thesis Statement 4
1.3 Overview of the Dissertation Studies 5
1.3.1 Evaluating the Reliability of Router Geolocation 5
1.3.2 Optimizations for More Accurate Continuous IP Geolocation 6
1.3.3 Identifying Cellular IP Blocks 7
1.4 Research Contributions 8

Chapter 2 Background and Related Work 11
2.1 Location Information from Existing Databases 11
2.1.1 Look-up *WHOIS* Databases by IP Address 11
2.1.2 DNS LOC Records 12
2.1.3 Location Hints within Domain Names 13
2.1.4 Public and Commercial Geolocation Databases 14
2.2 Target-Assisted Geolocation 15
2.2.1 Utilizing Technology-Enabled Devices 15
2.2.2 Crowdsourcing-based Geolocation 16
2.3 Measurement-based IP Geolocation 17
2.3.1 Terminology 17
2.3.2 Nearest Landmark to Target 18
2.3.3 Geolocation via Delay-to-Distance Mapping 20
2.3.4 Delay-based with Topology Geolocation 26
2.3.5 CBG Variations Comparison 32
2.4 Evaluating the Accuracy of Geolocation Services 32
2.5 Scale up of Measurement-based IP Geolocation 34
2.6 Cellular Block Identification 36
2.7 Conclusions 38

Chapter 3 Limitations of Router Geolocation in Popular Geolocation Services 40
3.1 Introduction 41
3.2 Datasets 43
3.2.1 CAIDA Topology Dataset 43
3.2.2 Geolocation Databases 43
3.3 Ground Truth Data 44

3.3.1 DNS-Based Ground Truth Data . . . 44
3.3.2 RTT-Proximity Ground Truth Data . . . 44
3.3.3 Ground Truth Data Regional and Topological Distribution . . . 45
3.3.4 Ground Truth Data Correctness . . . 46
3.4 Methodology . . . 49
3.4.1 Databases' Coverage and Consistency . . . 49
3.4.2 Same City Coordinates Across Databases . . . 50
3.4.3 Accuracy of the Databases . . . 51
3.5 Results and Discussion . . . 51
3.5.1 Databases' Coverage . . . 52
3.5.2 Databases' Consistency . . . 52
3.5.3 Evaluation Using Ground Truth Data . . . 53
3.6 Recommendations . . . 58
3.7 Conclusions . . . 60

Chapter 4 IP Blocks Co-locality . . . 62
4.1 Introduction . . . 62
4.2 Dataset Description . . . 64
4.3 Identification of Co-located IP Addresses . . . 65
4.3.1 Methodology . . . 65
4.3.2 Methodology Limitations . . . 67
4.4 Validating Identification of Multi-Location Blocks . . . 68
4.4.1 Building a Single-Location Ground Truth Dataset . . . 68
4.4.2 Building a Multi-Location Ground-Truth Dataset . . . 69
4.4.3 Validation . . . 70
4.4.4 Bounding the False Positives . . . 71
4.5 Co-Locality of /24 Blocks in the Wild . . . 72
4.5.1 Identifying Multi-Location /24 Blocks . . . 73
4.5.2 Characterizing Multi-Location /24 Blocks . . . 73
4.6 Identifying Arbitrary-Size Clusters of Co-located Addresses . . . 76
4.6.1 Similarity of Co-located Blocks Latency . . . 76
4.6.2 Evaluation Dataset . . . 77
4.6.3 Results . . . 78
4.7 Conclusions . . . 79

Chapter 5 Delay-based Identification of Internet Block Movement . . . 81
5.1 Introduction . . . 82
5.2 Datasets . . . 83
5.2.1 Latency Information from the USC Internet Outage Data . . . 83
5.2.2 Paths from the CAIDA UCSD IPv4 Routed /24 Topology Dataset . . . 84
5.2.3 Paths from the CAIDA Internet Topology Data Kit . . . 85
5.3 Methodology: From Block Latency to Block Movement . . . 86
5.3.1 Stable Estimation of VP-to-Block Latency . . . 87
5.3.2 Common Patterns in IP-Block Latency . . . 87
5.3.3 Identifying Block Movement from Latency Measurements . . . 89

5.4 Controlled Experiments with Synthetic Data . . . 90
5.4.1 Simulation of Block Movement . . . 90
5.4.2 Building a Dataset with Synthetic Movement . . . 91
5.4.3 ROC Analysis . . . 93
5.5 Evaluation with Real-World Data . . . 95
5.5.1 Applying our Method in the Wild . . . 95
5.5.2 Validating Block Movement with Historic Traceroutes . . . 96
5.5.3 Validating Block Movement with Transferred Blocks . . . 97
5.5.4 Movement over Time . . . 99
5.6 Conclusions . . . 100

Chapter 6 Delay-based Identification of Internet Cellular Blocks . . . 102
6.1 Introduction . . . 103
6.2 Datasets . . . 104
6.2.1 Latency Information from USC Internet Outage Data . . . 105
6.2.2 Supporting Datasets . . . 105
6.2.3 Labeled Data (Best-Effort Ground Truth) . . . 106
6.3 Identifying Block Type from Latency Estimates . . . 108
6.3.1 RTT Variation in Known Blocks . . . 108
6.3.2 Consistency of Variance in Latency in Each Block Type . . . 110
6.3.3 Methodology and Classification Thresholds . . . 111
6.3.4 Evaluation with the Test Labeled Data . . . 113
6.4 Identifying Block Types in the Wild . . . 114
6.5 Validation . . . 117
6.5.1 Classification Accuracy . . . 117
6.5.2 Sensitivity of Classification to IQR . . . 118
6.6 Conclusions . . . 119

Chapter 7 Future Work and Conclusions . . . 121
7.1 Future Work . . . 121
7.2 Conclusions . . . 123

Bibliography . . . 127

LIST OF TABLES

2.1 Summary of the constraint-based approach variations. 33

3.1 Location statistics and regional distribution of the DNS-based and RTT-proximity router interface addresses. 46

4.1 Building the synthetic multi-location dataset from single-location blocks. 70
4.2 Results of clustering university /24 address block to identify a false positives upper bound. 72
4.3 Top 10 countries sorted on their total number of /24 blocks in the dataset and the corresponding multi-location blocks percentage per country. 74
4.4 Top 10 ISPs sorted on their number of multi-location blocks, and their corresponding percentages of ISPs' total number of blocks in the dataset. 75
4.5 Overall results of clustering IP addresses in the university VP-compatible blocks dataset. 79

5.1 Accuracy from method to the university blocks. Dataset: 2018q4. 92
5.2 Accuracy for different delay-change thresholds from known movement. Dataset: synthetic. 93
5.3 Accuracy at different delay-change thresholds from known movement at *continent scale*. Dataset: synthetic. 94
5.4 Block movement from Internet-wide data. Dataset: 2018q4: threshold: 9%. 96
5.5 Validation of block movement with traceroute data. Datasets: 2018q4, ITDK, and CAIDA-topology. 97
5.6 Validation of block movement with transferred blocks. Datasets: 2018q4 and RIR reports. 97
5.7 Block movement from Internet-wide data. Dataset: 2019q1: threshold: 9%. 100

6.1 Rules to classify blocks by their median IQR. 112
6.2 Summary of classification results. 115
6.3 Top 5 organization with blocks identified as mixed. 115
6.4 Top 5 organization with blocks identified as strictly WiFi. 116
6.5 Top 10 organization with blocks identified as cellular. 116
6.6 Labeled test dataset blocks classification results. 117
6.7 TPR and FPR for the classification of the test data. 118

LIST OF FIGURES

2.1 Regional Internet Registries (RIRs) [65]. 12
2.2 Constrained multilateration using three vantage points. 21
2.3 Indirect estimation of the distance between a Web-based passive landmark and a target host [128]. 24
2.4 Street-level evaluation over academic, residential, and public Internet datasets [128]. 25
2.5 Octant positive and negative distance constraints. 28
2.6 Octant intermediate routers piecewise localization and the use of intermediate routers as additional passive landmarks [129]. 29
2.7 Octant landmarks calibration using spline line interpolation [129]. 30
2.8 Octant comparison to some previous techniques. Evaluated in North America (50 landmarks and 104 targets) [129]. 31

3.1 Identifying hops near a probe based on their RTT-proximity. 45
3.2 An example of two RTT-nearby probes. 49
3.3 Distribution of distances for same city coordinates across pairs of databases. 51
3.4 Databases pairwise distance distributions show at least 29% city-level disagreements for different vendors. 53
3.5 Databases *vs.* ground truth geolocation error. The number of addresses in each CDF is enclosed in parenthesis. 54
3.6 Country-level accuracy breakdown by RIR for ground truth. NetAcuity is the most accurate in all regions. 55
3.7 Databases' country-level accuracy is unreliable in most countries but NetAcuity is relatively consistent. 56
3.8 Databases *vs.* ground truth geolocation error breakdown by RIR. Only routers with city information are included. 57
3.9 Accuracy of the geolocation databases by dataset (DNS-based and RTT-proximity) for IP addresses in ARIN and RIPENCC 59

4.1 Results of applying the delay-based clustering to 99 2-block combinations. 72
4.2 Distribution of the number of clusters for all 1.41M /24 blocks in the ISI dataset. . . . 73
4.3 Two /24 address blocks with near-identical RTT fingerprints from USC /16 prefix. . . 77
4.4 Results of clustering 65 university groups of IP addresses in VP-compatible blocks. . . 78

5.1 Box plots and 5%ile RTT for sample /24 blocks. Dataset: 2018q4. 86
5.2 Example latency states for blocks showing routing change and movement. 88
5.3 Real blocks (dataset: 2018q4) combined to make a synthetic block with known movement (dataset: synthetic). 91
5.4 Distribution of distances in all 1,540 block pairs with known movement. Dataset: synthetic. 92
5.5 ROC curve showing true and false positive rates over blocks with known movement. . 94
5.6 ROC curve showing true and false positive rates over blocks with known movement within a continent scale. Dataset: synthetic. 95

5.7 Latency data for two inter-RIR transferred blocks. Dataset: 2018q4. 98
5.8 ICMP responses for block 69.94.100/24, showing similar behavior over all three months. 99

6.1 Examples of daily RTT variability for /24 blocks with different Internet access types during September 2017. 109
6.2 Distributions of median IQR for randomly selected blocks. 111
6.3 ROC curves to show the TPR *vs.* FPR of the classification results over the labeled training data. 113
6.4 ROC curves of classification results over the test data. 114
6.5 Median IQR and the daily IQRs variation by block type during September 2017. 119

Chapter 1

Introduction

Internet Protocol (IP) geolocation associates an IP address with a real-world location. IP geolocation is vital for many Internet *location-dependent* applications, as well as several network research topics. The geographic context that IP geolocation provides can help service providers to improve their services and their clients' online experience, and network researchers to understand various network phenomena.

Providers of Internet services use geographic context in numerous ways to manage and improve their services and to increase user engagement. For example, *geo-blocking* is widely used in Video on Demand (VoD) services to limit or block access to content based on users' location [40, 83]. These services are popular with hundreds of millions of subscribers and tens of billions of dollars revenues [33, 44]. Furthermore, service providers use geographic context to increase users' engagement since users are more likely to engage with geographically relevant content, potentially increasing revenues for online businesses [31, 103]. For instance, depending on the location of the user, service providers can dynamically customize their online content such as the language and local contact information, a news agency can display location-related news to its readers, and a search engine can return more relevant results when a user searches for a nearby shipping office or a restaurant. Other location-dependent applications include online fraud prevention, proximal server selection, targeted advertising, and spatial analysis of users' data.

IP Geolocation also provides valuable context to networking research. Many projects use geolocation information for several purposes, such as to study geographic deployment of Internet resources and their utilization [4, 20, 22, 42], study and visualize routing footprints to detect BGP threats [121], bind cloud data to a location [52, 94], estimate geographic presence of Autonomous Systems (ASes) [132], detect routing paths that experience detour-paths [112], and to study censorship and monitoring that happens in different countries [92]. All such studies

rely heavily on IP geolocation, often obtained from public and commercial geolocation services [5, 30, 32, 66, 80, 81].

1.1 Problems Addressed

In this section, we present the main problems and gaps we identify and address in this dissertation concerning IP geolocation.

Need to evaluate the reliability of router geolocation in geolocation services: The growing number of Internet applications and network research topics that benefit from geolocation services makes it proper to evaluate the reliability of these services. Quantifying the error margins and identifying regions where geolocation services fail can substantially improve the quality of the beneficiary services and the research studies.

Previous work on evaluating geolocation services focused on their overall accuracy [55, 63, 95, 113, 115]. Such work is biased towards evaluating endpoints geolocation since there are far more endpoints than infrastructure nodes in the Internet. This bias signifies the need for additional work to evaluate the geolocation of network infrastructure, such as routers, in commonly used geolocation services.

Need to enable efficient employment of the state-of-the-art geolocation methods: It is essential to enable the state-of-the-art geolocation methods, current or future, to scale up as services that maintain up-to-date location mappings for the whole Internet (§2.5). For example, the advantages of the measurement-based geolocation methods (§2.3) make them a desirable solution for continuous geolocation. These advantages include accuracy, the ability to estimate the current location with new measurement, provide a measure of confidence in the estimated location. Additionally, these methods are mostly user-independent (§2.3). However, measurement-based methods can be prohibitive to apply for both the source and destination networks [61], especially when applied frequently to all IP addresses to maintain up-to-date ge-

olocation for the entire Internet (§2.5). Full and frequent geolocation of the entire Internet is both expensive and unnecessary.

A better employment of the measurement-based geolocation methods would be to measure and geolocate only a few representatives from each group of co-located IP addresses, then assign the inferred location to the whole group. The primary challenge for this approach is to identify groups of co-located addresses before selecting their representatives. A /24 block might be an appealing choice. However, this choice might lead to substantial geolocation errors in blocks that span large geographic areas (§2.4). Also, a small unit size such as a /32 or a /24 block could waste chances of more efficient geolocation for cases of larger groups of co-located addresses. To avoid geolocation error caused by incorrect co-locality assumptions and to identify groups of adjacent addresses that we can geolocate as a unit, we need a method to access the co-locality of IP addresses.

Need to identify blocks that move: Only a fraction of the IP addresses move to a different location from time to time, for instance, when they are transferred to a different organization, or reassigned within an organization (Chapter 5). While we can maintain up-to-date location mappings with frequent geolocation updates for the entire Internet, such an approach would be expensive and unnecessary since only a fraction of the addresses may move from time to time.

A geolocation system needs to identify when a range of addresses moves to trigger a re-geolocation for these addresses. As a result, a geolocation system can maintain up-to-date location mappings without needing to perform full re-geolocation. Triggering re-geolocation only when it is needed helps the efforts to scale up continuous geolocation for the entire Internet.

Need to identify cellular blocks: More and more devices access the Internet through cellular networks. Cellular blocks are an interesting class of IP blocks to IP geolocation for several reasons. IP addresses in these blocks show distinctively large variation in latency when probed repeatedly from a vantage point (§6.3.1), which can affect the results of measurement-based

geolocation algorithms. Previous work has shown that the IP addresses in cellular networks are often shared across distinct geographic areas [11, 131]. Other work shows that popular geolocation services poorly geolocate the majority of IP addresses in cellular networks for which they have location ground truth data [124]. Identifying cellular blocks provides an additional layer of information, allowing for studying their geographic properties and for better interpretation of their delay observations.

1.2 Thesis Statement

The thesis of this dissertation states that **we can enable efficient, continuous IP geolocation by identifying clusters of co-located IP addresses and their location stability from latency observations.** In this statement, a cluster indicates a group of an arbitrary number of co-located adjacent IP addresses (a few up to a /16). Location stability indicates a measure of how often an IP block changes location. We gain efficiency by allowing IP geolocation systems to geolocate IP addresses as units, and by detecting when a geolocation update is required. These optimizations were not explored in prior work. Accurate IP geolocation algorithms that are prohibitive to apply to the entire Internet can use our optimization techniques to scale up as continuous services that maintain up-to-date geolocation. We present two studies to support this thesis statement. In the first supporting study, we present a delay-based method to identify clusters of *co-located* IP addresses by the similarity in their latency observations. In the second study, we present an efficient delay-based method to identify IP blocks that move. We show how these optimizations reduce the number of targets to geolocate while dictating when geolocation updates are required.

Additionally, we present a study to evaluate router geolocation in popular geolocation services, showing their limitations and the need for better solutions, an additional motivation for our work to enable the more accurate but expensive geolocation solutions. In our final study, we present another delay-based method that extends our use of latency measurement to characterize IP addresses. This time, we leverage latency observations to detect IP addresses in

cellular networks by their distinct variation in latency. This method allows for better interpretation of the latency observations of an interesting category of Internet blocks and to study their geographic properties without the need for proprietary data from carriers or users.

1.3 Overview of the Dissertation Studies

In §1.1, we presented some open problems and gaps in the IP geolocation body of work. In this section, we present an overview of the studies proposed in this dissertation to address these problems and how they support the thesis statement in §1.2.

1.3.1 Evaluating the Reliability of Router Geolocation in Popular Geolocation Services (Chapter 3)

Location-dependent applications and network research often rely on public and commercial geolocation services, typically provided as IP-to-location databases (§2.1.4). Previous work on the evaluation of geolocation services is biased towards evaluating endpoints geolocation [55, 63, 95, 113, 115]. This dissertation complements prior evaluation work by focusing on router geolocation in popular geolocation services.

We create a ground truth dataset of 16,586 router interfaces and their city-level locations using two different approaches, a DNS-based approach and a delay-based approach. We use our ground truth dataset to evaluate router geolocation accuracy of four geolocation databases by region. Our results show that the studied geolocation databases are not reliable for router geolocating and that there is room to improve both their country- and city-level accuracy.

Moreover, using Huffaker *et al.* domain-specific rules [64] (§2.1.3), we also show, in §3.3.4.2, that even router IP addresses move from time to time. These results show that we need better geolocation solutions and motivate our work to enable the more accurate but expensive solutions such as those discussed in §2.5.

1.3.2 Optimizations for More Accurate Continuous IP Geolocation

Earlier in the introduction, we discussed two optimizations to enable efficient, continuous IP geolocation that keeps location mappings up-to-date. In this section, we present an overview of our solutions to achieve these optimizations and how they demonstrate the thesis statement.

1.3.2.1 Identifying Clusters of Co-located IP Addresses (Chapter 4)

The first optimization we propose for efficient IP geolocation is to identify adjacent co-located IP addresses that we can geolocate as a unit, effectively reducing the number of targets without affecting accuracy. To achieve this optimization, we devise a delay-based method to identify adjacent co-located IP addresses by the similarity of their latency estimates.

We formulate this problem as finding similar IP addresses in a multidimensional space of delay coordinates. Adjacent, nearby IP addresses are expected to have small distances between them in the delay multidimensional space. We create for each IP address a vector of delay measurements observed from several vantage points. We then cluster IP addresses in a block based on the similarity of their delay vectors.

We assess the co-locality of IP addresses in a large subset of a latency measurement dataset collected by Hu *et al.* [61]. We first used our method to assess a previously common assumption that all IP addresses within the same /24 IPv4 prefix reside nearby (the *co-locality* assumption). Our results show that this assumption is incorrect for a significant fraction of the studied blocks. For such blocks, assigning one location for all the addresses in the block can lead to geolocation errors.

We then evaluate our algorithm on larger sets of adjacent IP addresses in 65 /16 university blocks that we believe are nearby. The results show that our algorithm can effectively identify arbitrary-size groups of co-located addresses. For most of the sets, the IP addresses are correctly identified in one cluster, indicating they are in one location. These results support our thesis statement by showing we can identify large clusters of co-located addresses that we can geolocate as a unit. Moreover, our method can help avoid geolocation error caused by making incorrect co-locality assumptions.

1.3.2.2 Identifying Blocks that Move (Chapter 5)

The second optimization reduces the burden of continuous geolocation by identifying when a geolocation update is needed, that is when an IP block change location. To achieve this optimization we propose a lightweight method to identify IP block movement by observing significant delay changes to /24 blocks from a few vantage points globally distributed. Delay changes can be a result of different reasons, including transient network congestion and routing changes. To minimize false positives, we only consider delay changes as strong evidence of movement when they are persistent and observed by several vantage points at different locations around the same time.

To quantify how often blocks move, we apply our algorithm to Internet-wide existing ICMP scan data §5.2.1 from publicly available measurements [97]. The results show that most of the responsive /24 blocks were RTT-stable over a quarter, 2018q4, suggesting location-stability. (We identified only 2.1% of the blocks as moving during this quarter.) We validate our approach by confirming movement through traceroutes and information about Internet registration reallocations.

These results show strong evidence that supports the thesis statement. Using an efficient method that identifies *movement of IP blocks* from existing ICMP scans, we show that only a small fraction of the IP blocks require geolocation updates. These results show that our method can help a geolocation service avoid frequent full re-geolocation to keep up-to-date location mappings.

1.3.3 Identifying Cellular IP Blocks (Chapter 6)

A large number of devices access the Internet through cellular networks today [37, 120]. Identifying IP blocks used for cellular access allows for a better understanding of network traffic trends and diagnose performance issues, among other applications. For IP geolocation, it is useful to identify blocks in cellular networks as they represent an interesting as well as a challenging category to geolocate without proprietary data (§2.6).

We propose a new method to identify cellular blocks based on active measurements of public IP address blocks. We show that we can distinguish between cellular blocks and other block types (fixed-line and WiFi) by the variance in their latency measurement when probed repeatedly from a vantage point. We measured the daily variance in a block's latency using the *interquartile range* of RTTs in a day, and then use the *median IQR* of all daily IQRs over a month to distinguish block type.

We show that cellular blocks exhibit different median IQR patterns to those of fixed-line and WiFi blocks. We also found that the median IQRs of fixed-line and WiFi blocks largely overlap, allowing only a fraction of the WiFi blocks to be identified with high confidence. As a result, we identified three categories (mixed, WiFi, cellular) of block type with two thresholds of median IQR, and selected good thresholds based on our labeled data, a form of best-effort ground-truth.

We apply our method to 3.72M /24 blocks and identify around 4.6% as cellular. We validate part of our algorithm predictions in the wild, showing high accuracy in cellular blocks identification.

This study extends our use of delay observations to characterize Internet addresses in ways that benefit IP geolocation and other networking applications. Cellular blocks show distinctive latency patterns which may affect the results of delay-based methods. Identifying these blocks provides a starting point for future work that studies their impact on delay-based methods. Our work will assist future work that study the geographic properties of cellular blocks and characterize how cellular networking affects the Internet.

1.4 Research Contributions

This dissertation presents four studies to address some challenges related to IP geolocation and to support the thesis statement. In this section, we list these studies and their main contributions.

The first study presents an evaluation of router geolocation in popular geolocation services, provided as IP-to-location databases (Chapter 3). We use a set of 1.64M router interfaces ex-

tracted from *CAIDA's Ark* dataset (§3.2.1) to study four popular databases inconsistencies and coverage. We show that the studied geolocation databases have many inconsistencies, especially at city-level. To assess the accuracy of the databases, we create a ground truth dataset using two approaches, a DNS-based approach (§2.1.3) and a delay-based approach that utilizes the RIPE Atlas built-in measurements [105]. Using these two approaches, we create a ground truth dataset of 16,586 interface IP addresses and their locations with city-level accuracy. We evaluate the databases' country- and city-level accuracy regionally using our ground truth dataset. Our results show that all studied databases have room to improve their accuracy, even at country-level. We provide a set of recommendations for using the geolocation databases to geolocate routers.

Our second study implements and evaluates a delay-based clustering algorithm to identify co-located adjacent IP addresses (Chapter 4). We first used our algorithm to identify blocks with endpoints at different locations automatically. We applied the algorithm to analyze 1.41M /24 blocks (118M addresses) and found that a noticeable fraction of these blocks (17%) appear to have endpoints at multiple locations. We then showed that our algorithm can identify clusters of co-located IP addresses of arbitrary size. This result shows that we can identify and geolocate large groups of co-located IP addresses as units, leading to a significant reduction in the number of targets and network traffic required for geolocation.

The third study defines a new delay-based method to identify IP blocks that move (Chapter 5). We used our method to quantify the fraction of moving blocks over two quarters (2018q4 and 2019q1) of Internet-wide existing ICMP scan data §5.2.1. We show that most of the responsive blocks are RTT-stable, suggesting their location did not change. We identified about 2.1% of the blocks as moving during 2018q4, and about 1.7% during 2019q1. A geolocation service can use our method to keep up-to-date geolocation without needing to perform frequent full re-geolocation for the entire Internet.

Our final study presents a new delay-based method to identify block type, mainly cellular IP blocks (Chapter 6). We show that we can identify cellular blocks by the variation in their RTT

observations. We applied our method to most of the public Internet, reporting on about 3.72M responsive IPv4 /24 blocks data from September 2017, identifying around 169k blocks (4.6%) as cellular. Our method identified most of the blocks as mixed (fixed-line or WiFi).

Chapter 2

Background and Related Work

Previous work on IP geolocation focused on proposing methods to identify the locations of IP addresses, evaluating the accuracy of existing geolocation methods and services, and the scale-up of measurement-based methods. In this chapter, we discuss the main methods to map IP addresses to real-world locations in terms of the approach (how the location is identified), accuracy (prediction *vs.* actual location), and coverage (addresses covered by the method). We also talk about the identification of cellular IP blocks, an interesting category to measurement-based geolocation given cellular blocks' distinctive latency patterns.

There are different methods to obtain IP-to-location information, including methods that infer location data from *existing databases* (§2.1), *target-assisted* methods (§2.2), and *measurement-based* methods (§2.3). In addition to these geolocation methods, we present prior work on evaluating the accuracy of IP geolocation services (§2.4). We then present prior work on the scale-up of measurement-based geolocation to the whole Internet (§2.5). Finally, we discuss prior work on the detection of blocks in cellular networks (§2.6).

2.1 Location Information from Existing Databases

In this section, we present several methods that infer location information from existing databases, including registry databases (§2.1.1, §2.1.2, and §2.1.3) and public and commercial geolocation databases (§2.1.4).

2.1.1 Look-up *WHOIS* Databases by IP Address

Public *WHOIS* databases store information about registered users of Internet resources, such as domain names and IP address blocks. Regional Internet Registries (RIRs) manage the *WHOIS* databases. Each RIR is responsible for a different region, as Figure 2.1 depicts (e.g., AFRINIC is the RIR for Africa). RIRs are responsible for dividing and delegating Internet address

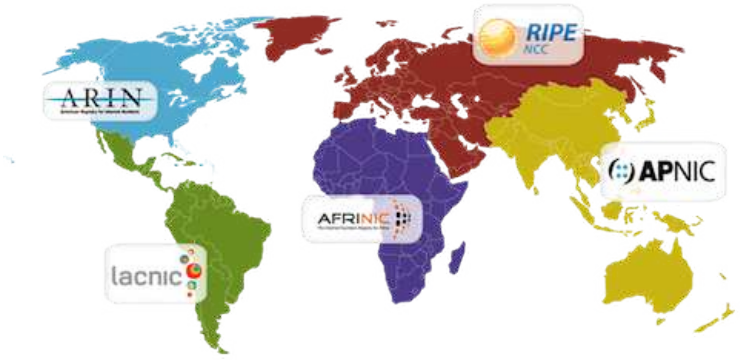


Figure 2.1: Regional Internet Registries (RIRs) [65].

blocks in their designated regions to the requesting entities (e.g., companies, organizations, and Internet service providers (ISPs)).

The process of registering Internet addresses involves maintaining the contact information of the registrant. The contact information can include location information such as the address of the registrant. Users can look-up *WHOIS* databases by IP address to obtain such contact information. *NetGeo* [84], and *GeoCluster* [91] are examples of earlier geolocation methods that rely on location information from *WHOIS* databases.

WHOIS databases have complete coverage of the allocated IP address space, but they can be inaccurate for IP geolocation purposes. The location information inferred from querying a *WHOIS* database often reflects the location of the registrant, which can be different from the location of the network itself. An entire IP prefix might be assigned a single location (e.g., the location where an organization is headquartered) even though the prefix is partitioned by its owner for use in different locations. Furthermore, the information in a *WHOIS* database can be outdated if an address block is moved without updating the database.

2.1.2 DNS LOC Records

Davis *et al.* in RFC 1876 [29] proposed adding location information about hosts and networks to the Domain Name System (DNS) via a new DNS Resource Record (RR) called *LOC*, which can be used to express the latitude, longitude, and altitude information.

Several geolocation methods tried to take advantage of this location information, including *GeoTrack* [91], *Gtrace* [93], and *VisualRoute* [126]. The deployment of this experimental protocol was hampered by the need to modify the *LOC* records by administrators who have little incentives to do so [91, 128]. Previous work has shown that most hosts lack LOC records [85]. Moreover, users who submit the information of the LOC records may intentionally provide incorrect data.

2.1.3 Location Hints within Domain Names

Some IP addresses map to domain names that include location hints. Decoding these location hints can reveal the location of such addresses [64, 77]. The Rocketfuel's *undns* tool was an early attempt to infer router location from information embedded in DNS names [6, 118, 119]. This work generated hand-crafted rule sets to parse names by conducting manual inspection of a list of router names to discover the naming conventions of different ISPs.

Freedman *et al.* added more name-parsing rules to *undns* to support more ISPs [43]. They used the tool to extract the locations of IP addresses as part of their work to study the geographic characteristics of IP prefixes and their influence on BGP routing tables. Their results show about 1.4% of /24 blocks or smaller span distances of more than 100 miles. (We discuss another geolocation method that uses the heuristics of the *undns* tool in §2.3.4.)

Inferring locations from DNS names can be challenging as network operators in different organizations may use different naming conventions. To address this challenge, Huffaker *et al.* propose a method to identify *domain-specific* naming rules, DNS-based Router Positioning (DRoP) [64]. They infer an extensive dictionary that maps location strings such as airport codes to physical coordinates, then using domain-specific rules they search for and decode location hints in hostnames to infer their locations. They generated domain-specific rules for 1,398 domains. Their work verifies inferred location hints using active latency measurement. Similarly, Scheitle *et al.* extract location hints from DNS names and then verify or disqualify them using latency measurement [110].

Related to extracting useful information from DNS names, Luckie *et al.* implement a method to identify a router name (router identifier) from hostname strings [78]. Their method depends on identifying naming conventions used by operators of different *suffixes*. (A suffix is defined as the part of a hostname that identifies an administrative domain, such as *comcast.net*.) Using training data of router interface IP addresses, router alias resolutions inferred with MIDAR [72] and Mercator [54], and the hostnames of interfaces, the method automatically generates regular expressions (*regexes*) to identify potential router names. A *regex* is concluded to define the naming convention for a given suffix if it identifies the same router name for all interfaces associated with the same router, such that the name is unique for all routers within the same suffix. This method identified the naming conventions, with confidence, for about 30.6% of 2,550 suffixes with data extracted from April 2019 CAIDA's ITDK IPv4 topology. This work can be useful for router geolocation as it provides another technique to identify interfaces on the same router that should all be geolocated to one location.

The DNS-based geolocation approach can provide accurate results. However, the scope of this approach is limited since not all router addresses have DNS names. Furthermore, not all names have useful geolocation hints.

2.1.4 Public and Commercial Geolocation Databases

IP geolocation services provide a popular, ready-to-use option to geolocate IP addresses. These services compile IP-to-location mappings as databases available as free [21, 30, 66, 81] or paid services [5, 32, 67, 80].

Providers of geolocation services often claim to have proprietary techniques to compile IP-to-location mappings. Lacking verified information about how these providers build their databases, we speculate that they may use different combinations of methods, such as: (a) parsing DNS names for location hints, (b) collecting data directly from users willing to share their location (e.g., via HTML Geolocation API [127] and MaxMind correction requests [82]), (c) collecting WiFi access points information including their MAC addresses and locations in association

with the devices that connect through them, (d) mining registry data, and (e) measurement-based geolocation. In §3.5.3.4, we show that one provider appears to benefit from location hints encoded in DNS names more than other competitors. As future work, we hope to investigate further how we can identify the methods providers use to compile their databases.

These geolocation services generally provide complete coverage for the IP address space at varying degrees of accuracy. Regardless of their accuracy, the location information these services provide could include geographic coordinates, postal code, city, and country information.

Prior work shows that such geolocation services are unreliable for geolocating end-hosts [49, 63, 75, 95, 113, 115]. We discuss the accuracy of geolocation services in §2.4. We also evaluate the accuracy of router geolocation in widely-used geolocation services in Chapter 3.

2.2 Target-Assisted Geolocation

In this section, we present two categories of methods that benefit from target hosts' collaboration. The first one benefits from certain technologies if enabled on target devices, such as the global positioning system (GPS) on a mobile phone (§2.2.1). The second one collects location data directly from users who are willing to share that data (§2.2.2).

2.2.1 Utilizing Technology-Enabled Devices

End-user devices may contain technologies useful to geolocate these devices with high accuracy. Global Positioning System (GPS), cell-towers triangulation, and WiFi-based geolocation are examples geolocation techniques that we can apply to devices that are GPS-enabled or connect via wireless means. These techniques typically require the permission of users to share their locations or to provide the required data to perform geolocation on their devices.

The GPS approach requires GPS-enabled devices to work. This approach provides the most precise geolocation, typically within a few meters of the actual location. Cell-towers and WiFi -based approaches, such as *Google Geolocation API* [53] and *Skyhook* [116], require a target device to be wireless-connected. These approaches perform triangulation based on the signals

and their strength between the target-host and the cell-towers (or the WiFi access points) [23, 125]. For these techniques to work well, they need the nearest cell-towers IDs or the host's visible WiFi access points. Triangulation using cell-towers provides accuracy within hundreds of meters and within a few tens of meters using WiFi access points [128].

These target-assisted methods are very accurate but have limited coverage. Devices that are not GPS-enabled or are not connected to a cellular network or a visible WiFi access point at a known location cannot be geolocated using such techniques.

2.2.2 Crowdsourcing-based Geolocation

Another source of IP-to-location information are users. Users may voluntarily provide their location information while doing online activities such as shopping. Lee *et al.* propose a crowdsourcing technique that utilizes an Internet-performance measurement-service, *Ookla Speedtest*, in Korea [75]. Users who use such performance measurement-services provide information that includes their *location*, which can be associated with their IP addresses. To increase the coverage of their method, they assign location information at an IP prefix level (/24, /25, and /26 prefixes). To improve confidence in the inferred locations, a majority vote of all locations gathered for IP addresses in a /24, /25, or a /26 is applied, requiring 80% or more location agreement to assign a location at prefix-level.

OpenIPmap is another project that relies, partially, on crowdsourcing input to map Internet Infrastructure, such as IXPs, and core routers, to geographic locations [106]. The project allows users to browse and submit geolocation information about network Infrastructure via a web application [107].

Reaching large counts of users for location information can be challenging, and many users are unwilling to provide such information. As a result, achieving high coverage is hard based only on information from participants in crowdsourcing efforts. Also, users may deliberately or accidentally provide incorrect location information. As a result, crowdsourcing is likely useful only as a complementary approach for geolocation.

2.3 Measurement-based IP Geolocation

Measurement-based IP geolocation relies on network measurements, mainly delay and topology, to geolocate an IP address. Delay measurements are mainly used to estimate the distance between two network nodes. While topology measurements can provide information about the intermediate nodes between a landmark and the target [70, 73, 129]. Both delay and topology measurements are mapped to constraints on the target's physical location.

Some geolocation methods use delay measurements to identify the closest landmark from the target IP address to associate its location with that of the landmark (§2.3.2). Most methods map delays to distances and use them to constrain the target location (§2.3.3). We expand our discussion of the geolocation methods that use delay-to-distance mapping to those that also incorporate topology (§2.3.4).

The measurement-based geolocation methods have several desirable advantages. First, previous work has shown that these methods provide good accuracy [56, 70, 128, 129]. Second, these methods are mostly *client-independent*; they only require the target address to respond to measurement. Third, they can be used to provide a *current estimation* of the target location with new measurement. Finally, these methods can provide a *measure of confidence* for their location estimation.

2.3.1 Terminology

We use the term vantage point (VP) to refer to an Internet host at a known location that can be used to actively probe other hosts. A vantage point is also referred to as an *active landmark* or simply a *landmark* in the literature. We use both, *VP* and *landmark* interchangeably throughout this chapter. In addition to active landmarks, some methods also use *passive landmarks* to refer to Internet hosts at known locations that cannot be used to issue probes to other hosts. We explicitly specify if a landmark is passive when that is the case, otherwise the term landmark indicates an active landmark. Active and passive landmarks are used by measurement-based methods to geolocate a *target IP address* or simply the *target*. Often, these methods estimate

the *great-circle distance* between two nodes, defined as the shortest distance between the two nodes on the surface of the Earth.

2.3.2 Nearest Landmark to Target

This section discusses methods that use delay measurements to find the potentially closest landmark from target. The goal is to associate the location of the target with that of the landmark. These methods do not translate delays into distance values, but they use them as an indication of distance or as signatures that identify co-located hosts as explained next in the discussion of these methods.

Shortest Ping

To estimate the location of target address, the *Shortest Ping* method probes the target from available VPs at different locations to find the VP with the *smallest RTT* to the target. The location of that VP is then assigned to the target IP address.

Several groups utilize Shortest Ping to study certain aspects of measurement-based geolocation. For example, as part of their work to scale up existing geolocation methods, Hu *et al.* use Shortest Ping to geolocate the entire IPv4 address space [61]. Also using this method, Eriksson and Crovella examine how network geometry affects geolocation accuracy [39]. Wang *et al.* use a modified version of Shortest Ping as the final stage of their geolocation system (§2.3.3.1) [128].

This method has two obvious limitations. The smallest RTT does not guarantee finding the closest VP to the target. Fore example, indirect paths and congestion can add to the observed latency making the VP looks farther than it really is from the target. And even if the algorithm always find the closest VP to the target, the accuracy is only as good as how far the VP is from the target. Finding the closest VP that is, for example, 200 km away from the target would likely results in incorrect city and possibly incorrect country geolocation.

GeoPing

GeoPing [91] is another method that seeks to identify the VP closest to a target. GeoPing is motivated by the insight that hosts with similar measured delays from the same reference points tend to be geographically close. GeoPing constructs an offline *delay-map* from *delay-vectors* of inter-VPs latencies. Each delay-vector contains one active VP position and the latency measurements from that VP to all other VPs:

$$(LM_X coordinates, delayLM_{1-X}, delayLM_{2-X}, ..., delayLM_{n-X})$$

The delay-vector can contain measurements to both, active and passive VPs (i.e., landmarks of known locations). Although passive VPs cannot be used to probe the target, it is possible to probe them to create additional delay-vectors and add them to the delay-map.

To estimate the location of a new target, a new delay-vector is constructed from latencies observed by probing the target from all VPs. The delay-vector of the target is then compared against those in the delay-map to find the VP with most similar delay-vector. The location of that VP is then assigned to the target.

Like the Shortest Ping method, GeoPing geolocation can be only as good as the closest VP from target, which could be too far. Moreover, previous work shows that the simpler Shortest Ping method does better than GeoPing in practice [38, 70].

Pattern-Based Geolocation (PBG) Approach

Pattern-Based Geolocation (PBG) is another single-point geolocation method that looks to find the closest landmark from the target [114]. PBG models geolocation as a pattern-recognition problem. Like GeoPing, PBG uses delays as signatures, but it builds these signatures statistically via Probability Mass Functions (PMFs). The intuition is that geographically close hosts have similar paths, therefore they also have similar RTT distributions, which can be used as signatures. PGP is designed specifically to improve the accuracy of geolocating IP addresses

in metropolitan areas where the authors indicate the delay-distance correlation is weak due to network congestion and higher queuing delays there.

To geolocate target addresses, the latency measurement is collected from probe nodes to passive landmarks and to the target to construct their individual PMFs. The target's PMF is then compared against those of the passive landmark using *Shifted distance divergence* metric to find the most similar one. This metric detects similar but shifted PMFs. Such shifts can be a result of few extra hops in one route. Finally, the location of the landmark with the highest similarity score is assigned to the target.

PBG has similar limitations to both *Shortest Ping* and *GeoPing*. Being able to find landmarks close from target is crucial to its accuracy. Statistically constructed signatures may achieve more robustness, but these signatures are not always strong enough to reveal the closest landmark to the target. The proposed solution in the paper adds significant amount of traffic.

2.3.3 Geolocation via Delay-to-Distance Mapping

This section discusses methods that map the observed delay between two nodes to a distance estimation. Mapping techniques include using a predefined conversion factor, such as the speed of light in fiber (measured to be around $\frac{2}{3}c$ where c is the speed of light) to estimated distance from delay. Another mapping technique builds a statistical model for the delay-distance relationship using measurements from VPs to targets at known locations. The model can then be used to predict the likelihood of a distance given a delay measurement.

Accurate delay-based estimation of the distance between two nodes can be tricky. Padmanabhan *et al.* admitted they were unable to capture delay-distance relationship using a precise mathematical model [91]. Distortion factors like circuitous paths and network queuing delays add to the observed delays, which complicates the delay-distance relationship. We next discuss different approaches that attempt to address delay to distance mapping challenges.

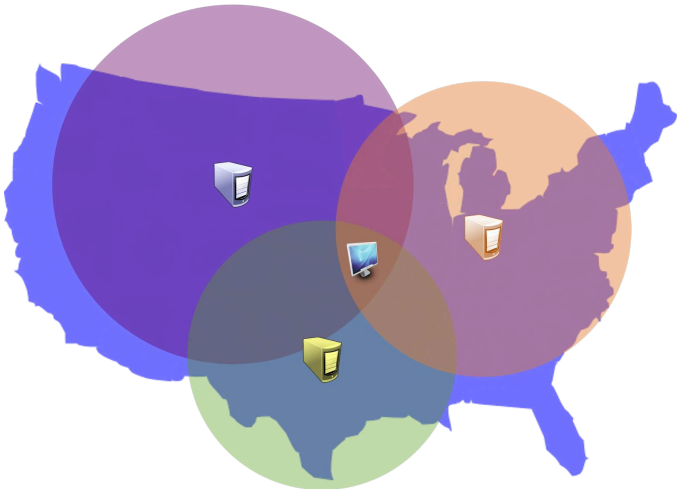


Figure 2.2: Constrained multilateration using three vantage points.

Constraint-Based Geolocation (CBG)

Constraint-Based Geolocation (CBG) maps a measured delay from a VP to the target into a region that constrain the target's location [56]. This region, hereby referred to as *location constraint*, is computed by first mapping the delay to a distance, for example using a conversion factor like the speed of light in fiber. Since delay measurements are often inflated, the estimated distance can be treated as an upper bound radius of a circle with the VP as its center, and the target is somewhere within its circumference.

The location constraint can be very large, therefore, we typically need multiple location constraints for more accurate geolocation. The intersection of these location constraints represents an estimation of the target location that satisfies all VPs constraints. Figure 2.2 shows an example of one target geolocation using three VPs.

There are several geolocation methods that utilize CBG. Some of these methods attempt to improve the accuracy of the delay-to-distance mapping, others try to include topology information to impose more restrictions on the location of targets. After explaining the basics of CBG here, we next turn to other CBG-based methods.

Bestline-Calibration Constraint-Based Geolocation

Gueye *et al.* are the first to apply the CBG approach to geolocate Internet hosts [56]. We refer to their method as *bestline-CBG* to avoid confusion with the basic CBG approach discussed earlier.

The intuition behind bestline-CBG is that the distortion in the relationship between network measured delay and distance can only be additive. In other words, the estimated distance between two hosts is equal to the great circle path length between them plus some error distance. This error is a result of additive delay terms, and the goal of bestline-CBG is to reduce this error.

In order to reduce the effects of delay-distance distortion factors on the estimated distance between two nodes, Gueye *et al.* proposed a landmark self calibration method. The method establishes a dynamic relationship between distance and network delays for each landmark. This relationship dictates how a landmark is calibrated to estimate the distance from an observed delay at a given time.

A landmark L_i is calibrated using the remaining landmarks. To establish a delay to distance relation, L_i measure the delay from itself to all other landmarks to create *(delay, distance)* data points. The *bestline* for L_i is defined as $y = m_i x + b_i$ such that it is closest but below all of its (delay, distance) data points. Each landmark needs to do the same to compute its own bestline. The bestline serves as a more aggressive delay-to-distance mapping when compared to the *baseline* computed as $\frac{1}{2} \times RTT \times \frac{2}{3}c$, where $\frac{2}{3}c$ is the speed at which bits travel in fiber and c is the speed of light.

Each landmark uses its *bestline* to compute the distance constraints from their measured delays to target. The constraints are then combined via multilateration to estimate a region for the target location. The size of the estimated region is an indication of the of the confidence in the geolocation.

There are several limitations that can affect bestline-CBG accuracy. First, sometimes the landmarks share part of the path toward a target hidden behind a single point. As a result, there

is insignificant gain from using the resulting constraints and the accuracy would highly depend on how far the closest landmark is from the target. Second, distance underestimation is possible due to the bestline calibration. Underestimating the distance by one landmark or more will result in an empty solution set. Third, the calibration approach is based on inter-landmarks network conditions at a given point of time. These conditions can quickly change rendering the calibration as futile. Furthermore, the inter-landmark calibration might not reflect the network conditions toward the target. Empirical results by Katz *et al.* [70] show no gains of using bestline-CBG method over a simpler CBG with no calibration.

2.3.3.1 CBG with Web-Based Landmarks

Measurement-based geolocation methods are shown to provide more accuracy as the distance between landmarks and targets decreases [61]. This can be attributed to less inflation in delay measurements compared to when landmarks are far away from the target, which makes delay and distance more correlated. Motivated by this observation, some approaches strive to find active and passive landmarks as close as possible to targets [61, 128].

One way to find landmarks closer from targets is to exploit *Web-based landmarks* such as Web or Mail servers hosted locally by universities, governments, and business entities at known locations. *Structon* and *street-level* geolocation are examples of methods that use web-based landmarks [58, 128]. The street-level geolocation method uses web-mining in an effort to find passive landmarks within a region around the target. This region is computed using the CBG approach.

To find web-based landmarks, the street-level method uses public services to find passive landmarks related to a set of ZIP codes. The set of ZIP codes is generated systematically based on the identified target's CBG region C_i. The extracted passive landmarks are then automatically validates to eliminate the incorrect ones (e.g., hosts that belong to businesses that use shared hosting services).

The Web-based passive landmarks are first used to tighten the region where the target is expected to reside in. This is done by estimating their distances to the target and then use

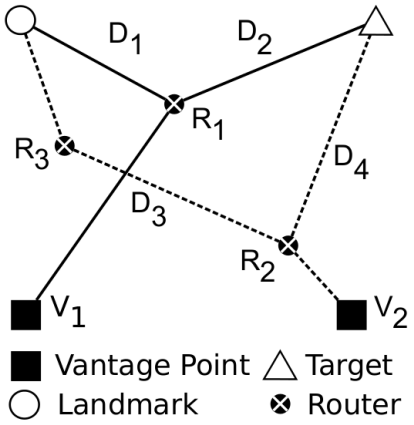


Figure 2.3: Indirect estimation of the distance between a Web-based passive landmark and a target host [128].

these distance to estimate a new constrained region C_{i-2}. To estimate the distance of a passive landmark to target, like landmark PL to target t depicted as an empty circle and a triangle respectively in Figure 2.3, each vantage point will *traceroute* both PL and t. The goal is to find the closest common router in the paths to PL and t. For vantage point v_1 this router is R_1, and for v_2 it is R_2. The latency from the common router is measured to both the passive landmark and the target. The summation of these two can then be used to estimate the distance between the landmark and the target. If different vantage points gave different estimations, then the minimum one is used.

Another possible approach to estimate the delay between the passive landmarks and the target is to use the *King* tool created by Gummadi *et al.* [57]. Gummadi *et al.* claim that the majority of hosts are close to their authoritative name servers. Based on that observation the tool is designed to estimate latency between any two hosts (even if we cannot access them) by estimating the *RTT* between their domain name servers using recursive DNS queries. The tool has its limitations though, for example end hosts that use modems have relatively large last hop latency.

To confine the location of the target furthermore, more web-based landmarks are identified within the new region C_{i-2}. The distances between them and the target are estimated as previ-

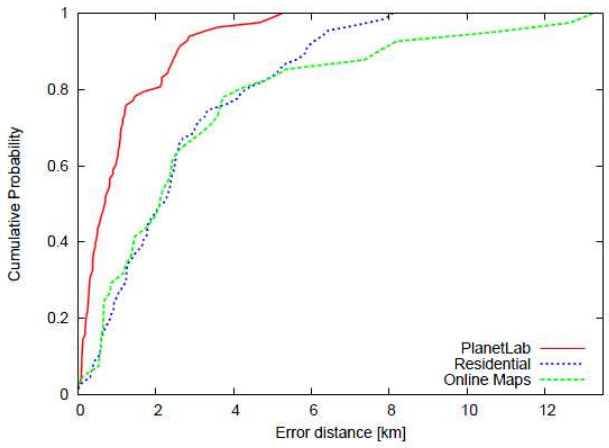


Figure 2.4: Street-level evaluation over academic, residential, and public Internet datasets [128].

ously described. The location of the relatively closest landmark to the target is then assigned to the target. This final solution is based on the observation that relative distances are preserved by delay measurements within a small scale.

The Street-level geolocation approach reportedly has the best results among all the geolocation approaches we discuss in this dissertation. A median error of about only 700 meters is reported. Figure 2.4 shows an evaluation of the approach using three different datasets to show how this method works in different networks environments; academic, residential, and the general Internet.

While the *street-level* method reported results shows high accuracy, the method still has several limitations. First, the accuracy of this approach depends on the landmarks density. Second, it is also less resilient to high variance latency connections close to end users as in the case of wireless networks. Third, the applicability of this method on a global scale is questionable, especially in regions that lacks means to enable discovery of useful passive landmarks. For example due to the lack of an addressing system equivalent to the postal codes used in the United States, or the lack of enough Internet resources that can be used as passive landmarks such as in rural areas. Finally, *street-level* method involves computation and traffic overhead, and even if the database of passive landmarks is created in advance, it still needs to be updated regularly.

2.3.4 Delay-based with Topology Geolocation

End-to-end delay-based geolocation approaches (e.g., bestline-CBG) are negatively affected by factors that inflate delay measurements such as networks congestion and path circuitousness. The accuracy of such approaches is shown to depend on the proximity of the landmarks from the target (the closer the better) [61, 70].

The Topology-based Geolocation (TBG) [70] and Octant [129] are two techniques that use topology information to improve upon end-to-end delay-based approaches. We discuss these techniques in this section.

Topology-Based Geolocation (TBG)

The *Topology-Based Geolocation* (*TBG*) approach uses network delay measurements and topology information to create a set of location constraints on the target host and the routers on the path from the landmarks towards the target [70]. Unlike end-to-end delay-based geolocation methods that only solve for the target location, TBG tries to solve simultaneously for both the target and intermediate routers locations.

Using inter-landmarks and landmarks-to-target *traceroute* probes, TBG infers per hop delays, end-to-end delays, and topology information. This information is used to globally optimize the topology picture (i.e., the locations of targets and intermediate routers) in accordance with the observed delays between all network elements.

To solve for targets and intermediate routers locations, TBG models the problem as a graph that contains targets, routers, and landmarks as nodes. The distance between two network nodes i and j is denoted as $d(i, j)$. Based on this model two types of constraints are defined:

- Hard Delay Constrains $(C_d) : d(l_i, x_j) \leq c_{ij}$
- Soft Link Latency Constraints $(C_l) : d(x_i, x_j) = h_{ij} + e_{ij}$

C_d indicates that the distance between a landmark l_i and a target or router x_j is bounded by the distance c_{ij}, the distance that light can traverse in fiber given the measured delay between the two nodes. C_l indicates that the distance between two adjacent nodes x_i and x_j is the

summation of two terms: h_{ij}, the estimated distance between two adjacent nodes based on their inferred hop latency, and the error e_{ij} resulted from the noise in the measurements.

Given the above constrains, TBG formulates the problem of geolocating targets as an optimization problem. The goal is to solve for the set of routers and targets (X) locations such that the observed distances between adjacent intermediate nodes correspond to hop delay measurements. The solution is required to minimize the total summation of the error term e_{ij} for all link constraints subject to C_d and C_l constraints:

$$\textbf{minimize:} \sum_{i,j \in C_l} |e_{ij}|$$

$$\textbf{subject to:} C_d, C_l$$

To improve estimations of intermediate routers locations, TBG clusters network interfaces that belong to the same router (i.e., a router's IP aliases). Two previously developed techniques are used to identify such IP aliases: Mercator [54], and Ally [119]. The clustering of a router interfaces helps observing tighter constrains on its location.

There are two more variations of TBG. The first variation uses passive landmarks to further constrain the network topology. TBG probes these passive landmarks from the active landmarks in order to add additional constraints on the location of intermediate routers between them. The second variation (in addition to using passive landmarks) uses location hints in the DNS names (§2.1.3). These hints are extracted via DNS parsing rules [119]. Given that the hints can be ambiguous or even incorrect, TBG validates these hints with latency measurements before using them.

TBG has several limitations. First, TBG may not work well for targets in networks that lack sufficient structural constrains. For example a stub network that communicates via one connection point with non-local hosts, especially when such point has a high latency towards targets within the stub network and all the landmarks are outside the network. Another issue is that optimizing for intermediate routers locations can lead to a greater error in estimating tar-

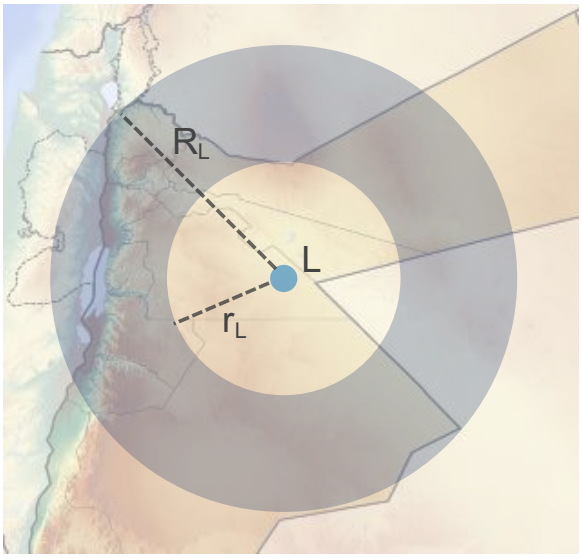


Figure 2.5: Octant positive and negative distance constraints.

get location. This can be a result of attempting to satisfy inflated delay measurements. Finally, TBG involves traffic and computation time overhead. This is a result of heavy use of traceroute probes to identify the network structure and to identify a router's IP aliases, and also due to solving for all network elements locations. This overhead is more problematic if the approach is to be applied to the wide public Internet repeatedly to address IP addresses location dynamics.

Octant

Octant is a geolocation framework that depends essentially on delay-to-distance constraints, but also allows for the inclusion of other types of constraints such as natural geographical constraints and demographic constraints [129]. Similar to TBG, Octant use both end-to-end and per hop delays to construct delay-to-distance constraints.

Octant uses both positive constraints (i.e., where the target is expected to reside) and negative constraints (i.e., where the target is not expected to reside). As a result, a landmark L will typically construct a constrained region that has the shape of a ring centered around L as Figure 2.5 illustrates. The positive and negative constraints for L are denoted by two radius values: R_L, the radius of the outer circle, and r_L, the radius of the inner circle.

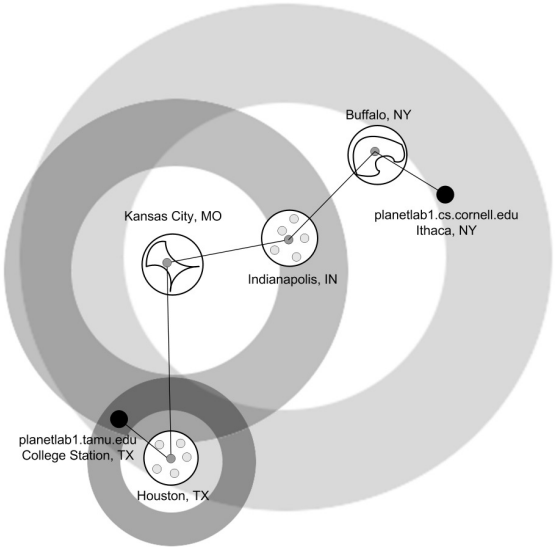


Figure 2.6: Octant intermediate routers piecewise localization and the use of intermediate routers as additional passive landmarks [129].

Unlike TBG's global optimization approach for locating targets and intermediate routers, *Octant* performs piecewise localization of routers on the path to the target. Each router is geolocated based on previous routers location estimations. Additionally, like TBG, Octant uses the *undns* tool to infer location hints from DNS names of the intermediate routers [6, 119].

The geolocated intermediate routers are used as secondary passive landmarks to add more constraints on the target location as depicted in Figure 2.6. Using the inferred delay between an intermediate router and the target, a delay-to-distance constraint is created based on the router's estimated location.

To improve the precision of the distance constraints, Octant applies a dynamic calibration process for each landmark. Octant measures latencies between a landmark being calibrated and the remaining landmarks. The data points in Figure 2.7 depicts landmarks known distances against measured latencies from a landmark L being calibrated. The two solid lines surrounding all data points compose a convex hull. The upper and lower lines correspond to the R_L and r_L values respectively. They correspond to all possible latency values for the calibrated landmark. These values are picked conservatively based on latency-distance data points. The R_L and r_L values are iteratively refined in order to tighten the estimated regions of targets later on. This is

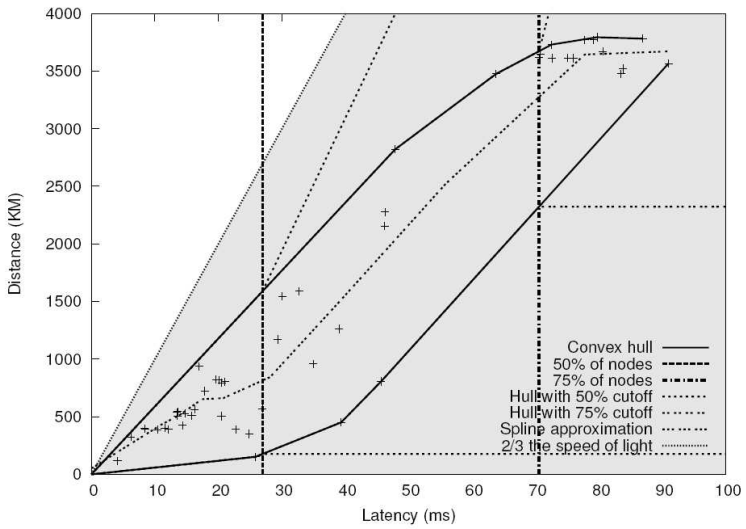


Figure 2.7: Octant landmarks calibration using spline line interpolation [129].

done by finding the interpolating spline line such that it minimizes the square error of delay-to-distance for the data points. A constant, δ, is chosen based on the percent of data points to be covered by the convex hull. The upper-bound is then computed by multiplying the spline with δ. The lower-bound constraint is computed by dividing the spline by δ.

As a results of combining positive and negative constraints, Octant's location estimates are potentially disjoint regions. These regions are separated by weights that indicate their likelihood. Figure 2.8 compares Octant's results on a PlanetLab [24] dataset to *GeoLim*, which refers to the CBG approach, and to GeoTrack, which uses the closest identifiable router to target as an estimation for its location. The reported median error for Octant is 35 km, a significant improvement over the other techniques.

The Octant framework for geolocation has several limitations. First, landmarks calibration reflects network conditions between landmarks, but not necessarily the landmark-target network conditions. Intermediate routers localization extracts information from routers DNS-names, which might be inaccurate or misleading and could cause errors in inferred typologies. Zhang *et al.* [134] report that 20 out of 182 (11%) edges in a Rocketfuel-like network [6] topology are actually false edges. Besides, this localization has city granularity, this coarse-grained granularity can subsequentially cause inaccurate constraints. Arif *et al.* [8] reported significantly

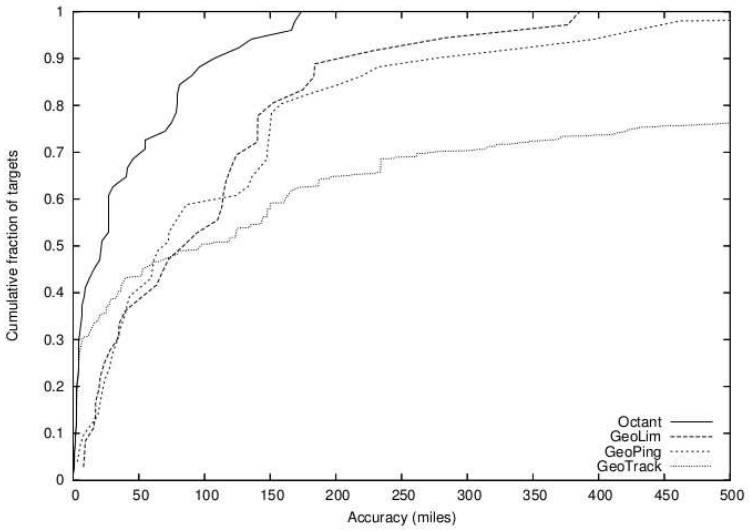


Figure 2.8: Octant comparison to some previous techniques. Evaluated in North America (50 landmarks and 104 targets) [129].

higher geolocation error results for their Octant implementation compared to those reported in [129]. They attributed that to the lack of clear minimum distance bound constraints in their evaluation dataset, and for not using some of the optimization techniques used by Octant.

TBG and Octant Comparison

Both approaches strive to overcome the problem of end-to-end measurements inflation by considering intermediate routers and hop latencies. TBG use these information to globally optimize for the locations of targets and intermediate routers such that they comply with observed network measurements. Octant geolocates intermediate routers, between a landmark and a target, serially, and uses them as additional (secondary) landmarks. Octant seems to be a more flexible approach that allows different types of constraints like geographic and demographic constraints to be incorporated. TBG, on the other hand, proposed to use several interesting techniques to infer further constraints on the topology picture, including the use and validation of location hints in DNS names, hop latencies and, the clustering of a router interfaces, and the use of passive landmarks.

Approaches such as TBG and Octant involve heavy use of traceroute measurements and require time to solve for routers and targets locations. Such methods can greatly benefit from our optimization techniques to reduce the number of IP addresses that need to be measured and geolocated (Chapter 4 and Chapter 5).

2.3.5 CBG Variations Comparison

Pure *end-to-end* delay-based approaches such as Shortest Ping, GeoPing, and bestline-CBG perform well when the landmarks are close from targets. However, their inadequacy to model circuitousness in routes becomes more prominent as the landmarks are farther away from the targets, as a result, their accuracy degrades. The accuracy of end-to-end delay-based approaches also degrades when landmarks share a high latency path towards target. Topology-based approaches, such as TBG and Octant, attempt to solve these issues by incorporating the intermediate routers in the process of estimating the target location. The *Street-level* method attempts to solve these issues by finding and exploiting landmarks closer to the target. While TBG, Octant, and *street-level* methods have higher accuracy compared to other end-to-end delay-based methods, they involve significantly more traffic and computation time. Table 2.1 gives a summary of the constraint-based variations reviewed in this section. Notice that the median errors in the table are reported from multiple previous studies when available.

2.4 Evaluating the Accuracy of Geolocation Services

Previous work on IP geolocation includes studies of the accuracy of geolocation services. These services are typically available as public and commercial *geolocation databases* (§2.1.4). In this section, we present several studies that focus on the geolocation accuracy of end-hosts in geolocation databases.

Several studies show that public and commercial databases have coarse-grained granularity and are not reliable at the city-level resolution. Some of these studies use delay-based methods to assess the reliability of the geolocation databases [49,55,56,115] Siwpersad *et al.* [115] studied

Table 2.1: Summary of the constraint-based approach variations.

Method	Landmarks	Use topology	Median error (km)	Evaluation region	Techniques
Bestline-CBG	Active	No	25 95	W. Europe US	-Per-landmark bestline calibration.
Street level	Active + Passive (Web-based)	No	2.1	US	-Extract and use web-based landmarks. -Relative latencies at small scale.
TBG	Active + Passive	Yes	67	N. America	-Global optimization of target and routers locations. -validation of hop latencies and location hints. -Network interfaces clustering.
Octant	Active + Passive	Yes	35	N. America	-Positive and negative constrains. -Geographic and demographic constraints. -Use routers as landmarks.

the geographic resolution of several geolocation databases. They compare location information provided by the databases with locations computed using Constraint-based Geolocation (CBG) [56]. They concluded that the resolution of the locations in the databases is generally way coarser compared to the geolocation results of CBG. Also using CBG, Gueye *et al.* estimated the max distance between endpoints of a block to estimate the block geographic span, which they conclude can be large [55].

Using a crowdsourcing technique to geolocate IP blocks in South Korea (§2.2.2), Lee *et al.* showed that one source of commercial geolocation databases inaccuracy is the assignment of one location to a whole large IP block [75]. They showed that MaxMind highly depend on *WHOIS* registry information. Moreover, they found typos in city names that matches typos found in in APNIC *WHOIS* registry. The study also showed that IP blocks locations changed over time but their geolocation was stable in the databases. In another regional study, Poese *et al.* suggested that databases can claim country-level accuracy but not city-level accuracy [95]. They examined the relationship between prefixes in several databases and those advertised by a large European ISP. They found that some databases split large ISP blocks into smaller ones for more accuracy but reported that the splitting did not improve the databases accuracy.

Other work examined different geolocation databases *coherency*. Huffaker *et al.* used majority vote across several databases to pick the location for a given block of IP addresses. They evaluated the databases based on majority-based selected location. Shavitt *et al.* [113] examined the coherency of geolocation databases using a ground truth dataset of IP addresses with known Points of Presence (PoP). Based on the assumption that IP addresses within the same PoP are co-located, they examine if the databases assign the same location for all the IP addresses in one PoP. Their results show a strong correlation between the databases.

While previous evaluation studies of the geolocation databases assess the overall accuracy of the geolocation databases, our evaluation work [48] in Chapter 3 focuses on assessing router geolocation in databases. Our ground truth is not specific to an ISP or region or PoPs as in [95] and [113]. We also do not use delay measurements as in [55] and [49] to study the geographic span and co-locality of IP blocks. Compared to [113], our work is different as we use a ground truth of IP addresses with known locations rather than co-locality information and a majority center of gravity to infer locations.

2.5 Scale up of Measurement-based IP Geolocation

Measurement-based IP geolocation can be prohibitive to apply to all IP addresses. We need to use several vantage points (VPs) to impose more constraints on the target location for better accuracy. We also need multiple latency measurements from each VP for better estimation of measurement noise. Moreover, the location of IP addresses can change as a result of reassignment to hosts at different locations, thus requiring continuous geolocation updates with new measurements. However, frequent geolocation of all IP addresses can be expensive, inefficient and unnecessary. Moreover, it could provoke complaints from the measured networks resulting in blacklisting of the VPs.

While several studies showed that measurement-based IP geolocation methods can provide accurate results (§2.3.4), few examined their application to the whole IP address space. Hu *et al.*

investigated scaling up a measurement-based method (*Shortest Ping*) to geolocate the entire IPv4 address space [61].

The study showed that the vantage points closest to target are the most important for the geolocation accuracy. The study also showed that selecting a subset of the landmarks based on their proximity to the targets improves scalability without adding much estimation error. They propose a vantage points selection algorithm for that purpose.

The vantage points selection algorithm is block oriented. The goal is to select the closest vantage points to each block of addresses (a /24 in this work). The selection is based on *RTTs* measured from all available vantage points to few responsive representative addresses within the block. The vantage points with the smallest RTTs are selected. Only the selected VPs are then used to gather measurements from all the addresses within that block. *Shortest Ping* method is used to evaluate the VPs selection algorithm.

A ground truth dataset of 18 /24 blocks and their location information is used in the evaluation. Each block has at least 100 responsive addresses. The addresses are geolocated using a set of 10 selected VPs. The addresses are also geolocated using all of the 400 VPs available. Results show almost identical estimation errors.

Our work addresses the challenge of scaling up measurement-based geolocation differently. We look to minimizing the number of targets to geolocate rather than minimizing the number of vantage points to use. Our IP block co-locality work presented in Chapter 4 uses IP addresses latency estimates as signatures to identify groups of similar adjacent endpoints. The goal is allowing automatic identification of clusters of IP addresses that can be geolocated as units.

We also propose a lightweight method to *identify if a block has moved* from one location to another, signaling the need to re-run an existing geolocation algorithm (Chapter 5). As a result, a geolocation system only need to re-run geolocation over the subset of blocks identified as moving. Unlike delay-based geolocation methods, our method works well with only a handful of vantage points regardless of their distance from the targets. We do not map latency estimates

to location constraints. Instead, we use them as fingerprints to dictate location stability. We are not aware of previous work that focuses on the identification of IP block movement.

2.6 Cellular Block Identification

IP address blocks in cellular networks represent an interesting class for IP geolocation. Previous work has shown that the IP addresses in cellular networks are frequently shared across many distinct locations [11, 131]. Other work shows that IP addresses in cellular network are poorly geolocated, reporting that around 70% of 29k cellular addresses across 50 countries are geolocated to 100 km or more from their actual ground truth location [124]. We do not know if these results are still relevant in today's cellular networks, but identifying these networks is a good step toward studying their geographic properties.

Several groups used client or operator information to identify cellular networks. Sen *et al.* implemented a framework requiring active participation from clients to study the performance of wide-area wireless networks [111]. To measure cellular networks performance, Nikravesh *et al.* used an application installed on users' mobile devices to collect active measurements [87]. In another performance evaluation work, Sommers *et al.* used crowdsourcing data from an interactive broadband speed tester (speedtest.net) to study the performance of 802.11 WiFi *vs.* cellular technologies [117]. Also relying on client collaboration, Rula *et al.* used data collected via two mobile applications installed on a few hundred mobile devices from major carriers in the US and South Korea [108]. This prior work uses "inside" knowledge to detect cellular networks, either from applications or operators. Our work (Chapter 6), identifies cellular IP blocks from external information (latency), without the need for clients or carriers collaboration. We do not study the performance of cellular networks in this dissertation, but our method to identify cellular blocks can help such studies in the future.

Recent work has used CDNs to identify cellular networks and characterize global cellular usage [109]. They infer the connection type of a device from the Network Information API, which allows access to information about the network connection a device is using [3]. The

data is collected by a large CDN monitoring system, which requires customer participation. Our approach does not require customer cooperation or the installation of any additional software at the client's end. We only require the IP addresses to respond to our probes.

Other work leverages information such as the browser and the operating system of devices accessing collaborating websites to identify their type (e.g., mobile or desktop) [34, 120]. Identifying mobile devices is useful but does not necessarily indicate a cellular connection since a device can have multiple interfaces that allow different connection types (e.g., WiFi, cellular, and Bluetooth).

Closest to our cellular block detection in this dissertation, Elmokashfi *et al.* studied the delay characteristics of 3 different Norwegian 3G networks [36]. They used multiple VPs to collect ping measurements. They found that different operators have different delay signatures. Moreover, these signatures are independent of the locations of the monitors. While this work studies delay characteristics of different cellular providers, it does not use RTT variation to identify cellular blocks. We also apply our approach more widely, to millions of networks and not just three ISPs.

Padmanabhan *et al.* analyzed ping latency from ISI Internet surveys [60] and ICMP data from ZMap [35] to estimate good timeouts for active probing [90]. They observed that ASes with most high latency addresses are cellular, but they did not provide methods to classify blocks blindly. Our results corroborate this observation. We also find that cellular networks exhibit high variation in latency and use that to identify them.

Finally, Cai *et al.* used full scans of /24 blocks to study IP block usage [12]. They used response patterns in edge hosts to identify blocks with low-bitrate access but did not explicitly identify wireless networks. They also examined block size, showing that /24 blocks are often used consistently. Our work uses the variation of RTT over time to identify patterns that distinguish between access types.

2.7 Conclusions

This chapter presented background and related work concerning the main approaches used to map IP addresses to physical locations and other IP geolocation related topics. We discussed the main proposed methods to identify the location of an IP address, we talked about work on evaluating the accuracy of existing geolocation methods and services, and touched on work about scaling up measurement-based methods. Finally, we talked about work related to identifying cellular IP blocks.

In our discussion of prior work, we identified several problems and future work directions concerning IP geolocation research. We summarize the main identified problems and gaps here along with pointers to the chapters that address them.

First, previous work that evaluated the geolocation accuracy of geolocation services focused on the geolocation of end hosts. We complement this work with our study of router geolocation in public and commercial geolocation services with a regional-breakdown analysis in Chapter 3.

Second, previous work related to IP geolocation did not focus on identifying geographically-homogeneous groups of IP addresses that we can treat as units. Some work focused on individual IP addresses, and some assumed that addresses in a block such as a /24 prefix are co-located. We propose a delay-based method to assess the co-locality assumptions and to identify arbitrary-size clusters of co-located addresses in Chapter 4. A geolocation system can use our method to identify IP addresses that we can efficiently geolocate together by a few representatives.

Third, we are not aware of prior work that identifies block movement. We propose a delay-based algorithm to identify movement in Chapter 5. Furthermore, we discussed, in §2.5, why measurement-based geolocation can be expensive to implement as a continuous service to the entire Internet. We show that the methods we propose to assess IP addresses co-locality (Chapter 4) and to detect address block movement (Chapter 5) address this challenge.

Finally, prior work on identifying cellular blocks uses "inside" knowledge to detect cellular networks, either from applications or operators. Our work in Chapter 6 implements an algo-

rithm that identifies cellular IP blocks from external information (latency), without the need for clients or carriers collaboration, a step toward studying their geographic properties and impact on delay-based methods.

Chapter 3

Limitations of Router Geolocation in Popular Geolocation Services

This thesis is about enabling more accurate geolocation methods as continuous services that scale up to the entire Internet, allowing for more reliable geolocation in comparison to widely used geolocation services. In this chapter, we present an evaluation of router geolocation in popular geolocation services as a compelling case for the need for more reliable geolocation and the need for continuous geolocation to maintain up-to-date IP-to-location mappings.

Internet measurement research frequently needs to map infrastructure components, such as routers, to their physical locations [22, 42, 52, 92, 112, 121, 132]. Although public and commercial geolocation services (often referred to as *geolocation databases*) are often used for this purpose, their accuracy when applied to network infrastructure has not been sufficiently assessed. Prior work focused on evaluating the overall accuracy of geolocation databases, which is dominated by their performance on end-user IP addresses (§2.4). In this chapter, we evaluate the *consistency* and *reliability* (*coverage* and *accuracy*) of *router* geolocation with popular geolocation databases (§3.2.2).

We use a dataset of about 1.64M router-interface IP addresses (or simply *router interfaces*) extracted from the CAIDA topology datasets [16] to examine the coverage and consistency of several databases at country- and city-level resolutions. Where coverage indicates the fraction of IP addresses with a database-location at a given level—regardless of location correctness—and consistency is a measure of agreement among databases at a given level. We also create a ground-truth dataset of 15.6k router interfaces and their country- and city-level locations (§3.3). We use this dataset to evaluate the databases' accuracy at both country- and city-level resolutions with regional breakdown analysis. Where the country-level accuracy of a database indi-

cates the fraction of addresses with the same country as the ground truth, and the city-level accuracy indicates the fraction of addresses within city-range (§3.4.2) of the ground truth.

Evaluating the reliability (coverage and accuracy) of the studied databases shows that they can be unreliable to geolocate routers at both country- and city-level resolutions. The database with the best overall results shows near-perfect coverage at both resolutions (have country and city geolocation for almost all addresses in the 1.64M router-interfaces dataset (§3.5.1)). However, this database geolocates only 89.4% of the ground-truth addresses correctly at country-level, and only 73% at city-level—within 40 km of the answer in the ground truth— (§3.5.3.1). The other databases are less accurate at both country- and city-level and two of them have significantly less coverage at city-level. These results show that the databases could lack extensive city-level coverage and can be inaccurate for geolocating routers at both country- and city-level, corroborating previous work on evaluating the overall accuracy of geolocation databases (§2.4). Moreover, we show that IP addresses assigned to routers can experience location change over time (§3.3.4.2), suggesting the need for re-geolocation. All of these results motivate our work to enable continuous, more reliable geolocation (Chapter 4 and Chapter 5).

3.1 Introduction

Networking research that needs to map Internet resources to their location often uses geolocation services for that purpose. Examples of research studies that use geolocation services include studying the geographic deployment of Internet resources [22], studying routing phenomena to detect BGP threats [121], estimating the geographic presence of Autonomous Systems [132], detecting routing paths that experience detour-paths [112], and studying censorship and monitoring [92].

These studies rely heavily on the accuracy of geolocation services especially for IP addresses that are used for Internet infrastructure (e.g., routers, switches). Quantifying error margins and identifying regions where geolocation services fail can substantially improve the quality of such studies. Geolocation services are typically available as third-party databases, publicly avail-

able [30, 66, 81] or paid [32, 67, 80]. Delay-based geolocation, where delay measurements are mapped to location constraints [56, 70, 91, 128, 129], is another viable option, especially with more public measurement platforms becoming available [35, 50, 105]. However, many users might still prefer the available ready to use geolocation databases.

Given the importance of router geolocation in understanding geographic aspects of the Internet infrastructure, our work focuses on router geolocation in both public and commercial databases. Previous work on evaluating databases focused on their overall accuracy [55, 63, 95, 113, 115]. However, such work is biased towards evaluating endpoints geolocation since there are far more endpoints than infrastructure in the Internet.

Researchers who use geolocation databases to learn routers' locations need to know how reliable are these databases in terms of their country- and city-level coverage (the fraction of addresses a database has country- and city-level resolutions for, respectively), and their accuracy across the world. In this chapter, we study router geolocation in four popular geolocation databases, two of which are free: MaxMind GeoLite2 [81], and IP2Location DB11.Lite [66], and two are commercial: MaxMind GeoIP2 [80], and Digital Envoy NetAcuity [32]. We explain why these databases are selected in §3.2.2.

Our main contributions in this chapter are: (1) we show that the studied databases have many inconsistencies for router geolocation, especially at city-level. We use a set of 1.64M router interface addresses extracted from CAIDA's Ark dataset (§3.2.1) to study all 4 databases inconsistencies and coverage, (2) we create a ground truth dataset[1] of 16,586 router interface addresses and their locations with city-level accuracy. We create our ground truth using two approaches, a DNS-based approach proposed by Huffaker *et al.* [64] and a delay-based approach that utilizes the RIPE Atlas built-in measurements [105], (3) we use the ground truth dataset to evaluate the databases' country- and city-level accuracy regionally. The results show that all the

[1]Our ground truth data is available via IMPACT: `https://www.impactcybertrust.org/dataset_view?idDataset=792`

databases have room to improve their accuracy, even at country-level; (4) our final contribution is a set of recommendations for using the geolocation databases to geolocate routers.

3.2 Datasets

This section presents an overview of the CAIDA topology dataset we use to extract our evaluation dataset of router interfaces (§3.2.1), and the geolocation databases we evaluate in this work (§3.2.2).

3.2.1 CAIDA Topology Dataset

We use the CAIDA topology dataset [16] collected using CAIDA's Ark measurement infrastructure. Ark monitors around the world collect traceroute data for randomly selected IP addresses from all routed /24 IPv4 prefixes. Using one week of the topology dataset starting from March 9, 2016, we extract a dataset of about 1.64M router interface IP addresses, which map to an estimated number of 485k distinct routers according to CAIDA's ITDK alias mapping results [17]. We treat this dataset at IP-level since the geolocation services are supposed to geolocate all IP addresses regardless of their alias resolution. We refer to this dataset as the *Ark-topo-router* dataset. We use this dataset to evaluate the country- and city-level coverage and consistency across the geolocation databases.

3.2.2 Geolocation Databases

We compare and assess router geolocation reliability in four popular geolocation databases: MaxMind GeoIP2 (referred to as *MaxMind-Paid* in this chapter), MaxMind GeoLite2 (referred to as *MaxMind-GeoLite*), IP2Location DB11-Lite (referred to as *IP2Location-Lite*), and Digital Element NetAcuity (referred to as *NetAcuity*). We chose the NetAcuity and MaxMind commercial databases as they are widely considered among the leaders in the geolocation business [63,113]. On the other hand, comparing the free and commercial versions of MaxMind's databases provides a measure of the improvement between the two. Finally, the IP2Location database is

known for providing city-level resolution for most of the IP address space and it appears often in geolocation comparative studies.

3.3 Ground Truth Data

Our ground truth data is basically a set of router interface addresses and their locations at city-level accuracy. We extract our ground truth using two methods as explained next. We discuss the correctness of the ground truth data in §3.3.4.

3.3.1 DNS-Based Ground Truth Data

To create the first part of our ground-truth, we use the DNS-based geolocation method from the work of Huffaker *et al.* [64] (see §2.1.3). The original work generated domain-specific rules for 1,398 domains, but we only use 7 domains for which the authors have ground truth-rules from the domains' operators (*ground-truth-domains*).

Performing reverse DNS (rDNS) lookups to the Ark-topo-router IP addresses on May 15, 2016 results in 905k IP addresses with hostnames. About 13.5k of these IP addresses belong to the ground-truth-domains of which we are able to geolocate 11,857 IP addresses as follows: *belwue.de* (23 addresses), *cogentco.com* (6,462), *digitalwest.net* (29), *ntt.net* (2,331), *peak10.net* (170), *seabone.net* (1,405), and *pnap.net* (1,437).

3.3.2 RTT-Proximity Ground Truth Data

We use RIPE Atlas *built-in measurements* [105] to create our second part of the ground truth. These measurements are issued by most of the probes toward well known targets, such as DNS root servers. We use traceroute measurements collected on May 25th, 2016. The measurements are provided in *JSON* format that specify the measurement origin, target, intermediate hops and their observed RTTs. Figure 3.1 illustrates our method to identify hops near to a RIPE Atlas probe based on RTT measurement. Since a 0.5 ms RTT between two locations maps to a distance of at most 50 km—likely less than that due to inflation in RTT measurement— we

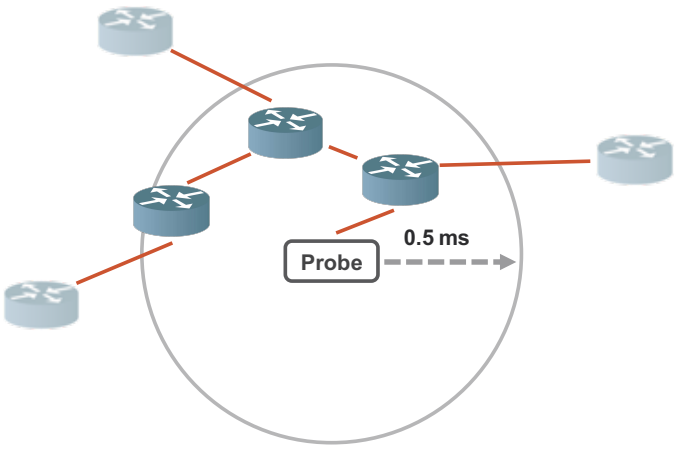


Figure 3.1: Identifying hops near a probe based on their RTT-proximity.

use 0.5 ms threshold to find all the hops guaranteed to be within 50 km of their probes (shown within the circle in Figure 3.1). We associate such hops with their probe's location.

We find 4960 router interface IP addresses that satisfy our 0.5 ms threshold but we only keep 4838 addresses due to the reasons we explain in §3.3.4.3. We refer to the set of 4,838 IP addresses and their locations as the *RTT-proximity* ground truth dataset. Note that while some of the gathered IP addresses could belong to home routers, more than 80% are at least 2 hops away from their probes indicating otherwise.

3.3.3 Ground Truth Data Regional and Topological Distribution

Table 3.1 shows statistics for our two ground truth datasets including the total number of addresses (column 2), number of unique countries where the addresses are located (column 3), number of unique coordinates (column 4), and the number of addresses found in each regional Internet registry (RIR) (columns 5 to 9). The RIR for each address is learned from querying Team Cymru *WHOIS* database [122]. According to CAIDA'S AS rank [15], transit ASes announce 74.5% of addresses in our RTT-inferred ground truth set and 99.9% of addresses in our DNS-based ground truth set.

Table 3.1: Location statistics and regional distribution of the DNS-based and RTT-proximity router interface addresses.

Ground Truth	Total	Countries	lat/lon	ARIN	APNIC	AFRINIC	LACNIC	RIPENCC
DNS-based	11,857	53	238	9,588	560	0	0	1,709
RTT-proximity	4,838	118	1,347	1,123	372	131	52	3,160

3.3.4 Ground Truth Data Correctness

In this section, we validate part of the DNS-based ground truth with RTT data from two other datasets (§3.3.4.1). Part of our DNS-based data validation examines identifying IP addresses movement from changes in their DNS names (§3.3.4.2). Finally, to increase the confidence in our RTT-proximity ground truth, we use two methods to disqualify RIPE Atlas probes that appear to have inaccurate geolocation (§3.3.4.3).

3.3.4.1 DNS-Based Data Correctness

We validate part of the DNS-based dataset using two latency measurement datasets: our RTT-proximity dataset and another similar dataset provided to us by Giotsas *et al.* [51]. Despite the small intersection between the datasets, we see positive signs indicating the correctness of the DNS-based data as we explain next.

First, we examine the common IP addresses between our RTT-proximity dataset and the DNS-based dataset. We identify 109 common addresses between the the two ground truth datasets. These two datasets agree within 10 km on the locations of 105 of the addresses and within 43 km on the remaining 4 addresses. These results show that the RTT-proximity dataset confirms the location of all common IP addresses with the DNS-based dataset. Next we examine common IP addresses between the DNS-based dataset and another RTT-proximity dataset.

Similar to the method we use to create our *RTT-proximity* ground truth (§3.3.2), Giotsas *et al.* looked for RIPE Atlas probes within 1 ms from a set of router interfaces of interest. Associating each router-interface with the location of the closest probe within 1 ms creates a router-to-location dataset (we refer to as the *1ms-RTT-proximity* dataset). The 1ms-RTT-proximity dataset has about 20.5k router interfaces, but only 384 are common with our DNS-based dataset. Pair-

wise comparisons of common IP addresses show that the locations from the DNS-based and the 1ms-RTT-proximity datasets are within 100 km for 355 addresses (92.45%). Given the 1 ms threshold used to create the 1ms-RTT-proximity dataset, these locations are fairly compatible; in fact, the locations of 337 (87.8%) addresses are within 40 km. The 1ms-RTT-proximity dataset was gathered about 10 months after our DNS-based dataset and some IP addresses could have moved during this period (§3.3.4.2), but overall, this dataset largely confirms the location of most of the addresses common with the DNS-based dataset.

3.3.4.2 Identifying Movement from DNS Names

Interestingly, out of the remaining 29 IP addresses with incompatible locations, we find that 19 addresses are likely reassigned to hosts at different locations. We observe this change from their rDNS records. For example, the rDNS lookup result for one IP address was *ae-5.r23.dllstx09.us.bb.gin.ntt.net* on May 2016 and *ae-3.a01.miamfl02.us.bb.gin.ntt.net* on September 2017. The location hint in the prior one indicates the location *Dallas, TX*, while the later indicates *Miami, FL*. All 19 addresses would have similar geolocation to that in the 1ms-RTT-proximity dataset given their updated hostnames, that said, we do not know when exactly the hostnames have changed and if that happened before creating the 1ms-RTT-proximity set. The location disagreement for some of the remaining 10 addresses might be a result of reassigning addresses to hosts at a different location without updating their hostnames leading to misleading location hints. Few RIPE Atlas probes may also have incorrect geolocation.

Overall, between May 2016 and September 2017, 8,197 (69.1%) of the 11,857 DNS-based addresses kept the same hostnames, 2,848 (24%) have different hostnames, and 6.9% no longer have rDNS records. Not all hostnames changes indicate location changes. Geolocating the 2,848 IP addresses with different hostnames using DRoP's domain-specific ground-truth rules shows that 1,927 (67.7%) still have the same location, 877 (30.8%) have different location—i.e., 7.4% of all DNS-based addresses in about 16 months—, and 44 (1.5%) no longer have location hints that match any of the rules.

3.3.4.3 RTT-Proximity Data Correctness

The correctness of the RTT-proximity data is dependent on the accuracy of the RIPE Atlas probes locations, which are mostly crowdsourcing-based. While the probes' hosts can easily provide correct city-level locations, it is not guaranteed that they always do. Additionally, a probe might be moved without updating its public location. RIPE Atlas operators informed us that they do some manual checking but nothing structural to validate probes' locations. To increase the confidence in the RTT-proximity data we use two methods to disqualify probes that appear to have inaccurate geolocation.

First, we identify and remove all probes assigned *default* country coordinates. These coordinates are typically near the geographic center of a country [25, 45, 79] and are often located in unpopulated areas (e.g., *N*51°00′00″ *E*09°00′00″ in Germany). Such coordinates are often assigned to IP addresses due to the lack of specific location information. From the set of 1,387 probes associated with our 0.5 ms threshold data, we find 19 probes within 5 km of their known country coordinates. Using traceroute measurements we are able to prove that many of these probes indeed have bad geolocation. We find and remove 109 IP addresses associated with these probes.

Our second method is based on the insight that multiple probes near the same router should also be near each other. We call such a group of probes RTT-*nearby* probes. Given our 0.5 ms threshold, any two RTT-*nearby* probes should be within a distance of 100 km. Figure 3.2 illustrates an example of two probes within 0.5 ms from the same router and therefore are RTT-*nearby* probes. We find 495 addresses in the remaining RTT-proximity data with RTT-*nearby* groups of 2 or more probes, out of which, only 12 addresses (2.4%) have RTT-*nearby* probes with inconsistent locations. For example, two probes in Mozambique are RTT-*nearby* to an IP address but their locations are 867 km apart, which means at least one of them has incorrect geolocation. We find 3 other RTT-*nearby* groups that have prominent location inconsistencies. The 8 remaining addresses have relatively small disagreements of less than 128 km between any two probes in one RTT-*nearby* group. One probe in Italy is responsible for 7 of those location

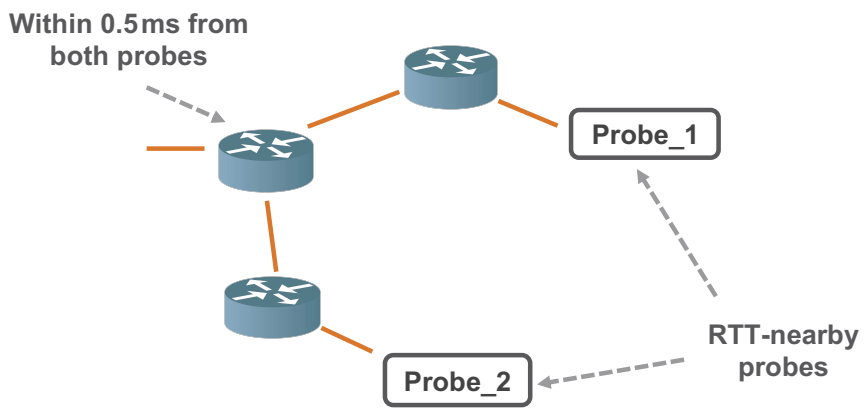


Figure 3.2: An example of two RTT-nearby probes.

disagreements. Overall, we have 223 different probes that are part of one or more RTT-*nearby* groups, out of which, we only disqualify 5 probes (2.2%) and remove 13 interface addresses associated with them. As a result, the final RTT-proximity dataset has 4,838 addresses.

We match the RTT-proximity and the 1ms-RTT-proximity datasets and find an intersection of 1,661 addresses. Comparing the locations from the two datasets for each common IP address shows that 96.8% and 97.4% of the addresses agree within 40 km and 100 km respectively. The small fraction with location disagreements might be a result of IP addresses reassignment to hosts at different locations during the time separating the two datasets.

3.4 Methodology

In this chapter, we seek to answer these questions: (a) what is the probability to find an answer for a router address geolocation query and what would be the resolution of the answer? (b) how consistent are the answers across different databases at both country- and city-level resolution? (c) what is the probability that the database answer is correct? We next explain how we answer these questions.

3.4.1 Databases' Coverage and Consistency

To evaluate the coverage and consistency of the participating databases, we use the Ark-topo-router dataset (§3.2.1). To evaluate a database coverage we find the percentage of ad-

dresses with location information in each database for both country- and city-level. We also evaluate the pairwise consistency at both resolutions.

While country-level consistency evaluation is as simple as comparing standard ISO alpha-2 or alpha-3 country codes in databases, the city-level consistency evaluation can be tricky, in part because different databases may use different city names. Rather than comparing city names, we compute the distance between one IP address locations (coordinates) in any two databases and check if it is within city range. Comparing coordinates invokes two questions: (a) does the database provide correct city-level coordinates for a given city in a location record? (2) what radius is acceptable as a city range?.

We compare each database coordinates for a given city with the city coordinates from a third party geographical database called *GeoNames* [45]. Since multiple cities can have the same name, we also include the region and country in the matching process. We observe that the distance between city coordinates from any of the geolocation databases and GeoNames is within 40 km more than 99% of the time, indicating that the databases are indeed assigning city-level coordinates when a city name exists in the location record.

3.4.2 Same City Coordinates Across Databases

Answering the question about city range is tricky, mainly because different cities can have drastically different areas. Previous work [113] used 40 km as their city range, while [63, 64] used the same distance as their threshold to identify if two locations are co-located. However, we note that different databases may assign different coordinates to the same city. We examine the distance between coordinates assigned to the same city across the databases, and find that one city coordinates from any two databases are more than 99% of the time within 40 km. We conclude it is reasonable to consider any two databases' coordinates within 40 km to be within the same city circumference.

Figure 3.3 shows the pairwise distribution of distances computed between one city location(s) in one database and the same city location(s) in another database. We identify each city

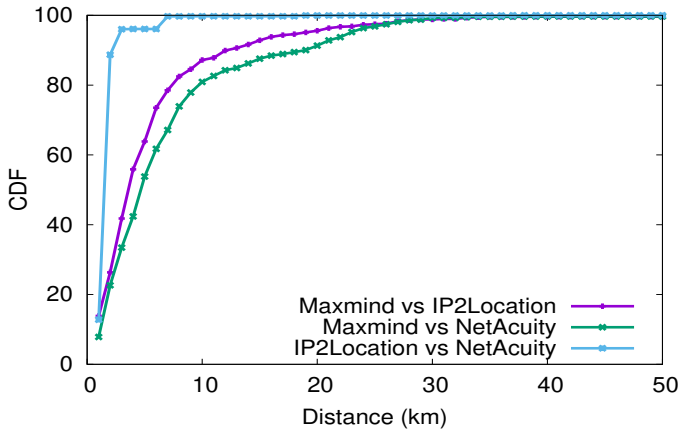


Figure 3.3: Distribution of distances for same city coordinates across pairs of databases. We observe that one city coordinates from any two databases is more than 99% of the time within 40 km.

by its name and country code, and the region/state. We compute the distance for all possible combinations for all the locations we find for a given city in all databases. MaxMind uses same city names across their paid and free version, hence we only use MaxMind-Paid in this analysis. From the figure we observe that the vast majority of the location pairs are at a distance of 40 km or less across databases, we conclude it is reasonable to consider any two databases' coordinates at such distance or less to be within the same city circumference.

3.4.3 Accuracy of the Databases

Finally, we evaluate the overall accuracy of geolocating routers in databases using our ground truth of 16,586 interface addresses with city-level accuracy. We also evaluate the accuracy by region where we breakdown the ground truth addresses by their RIRs and report the results for each region at country- and city-level (§3.5.3.2).

3.5 Results and Discussion

We first discuss the results of evaluating databases' coverage (§3.5.1) and consistency (§3.5.2) at country- and city-level over the Ark-topo-router dataset. We then discuss the results of evaluating databases' coverage and accuracy over the ground truth data (§3.5.3).

3.5.1 Databases' Coverage

Using the Ark-topo-router dataset, we analyze databases router geolocation coverage and consistency at country- and city-level. All the databases are accessed shortly after creating the Ark-topo-router set to geolocate its addresses. We find that IP2Location-Lite and NetAcuity both provide near-perfect coverage for all interface addresses in the Ark-topo-router dataset at both country- and city-level. The MaxMind-GeoLite and MaxMind-Paid databases both cover about 99.3% of the addresses at country-level, but only 43% and 61.6% of the addresses at the city-level respectively.

3.5.2 Databases' Consistency

Pairwise country-level comparison shows that the MaxMind databases agree on the location of 99.6% of the 1.64M interface IP addresses, while all other pairwise comparisons' agreements range between 97.0% and 97.6%. The overall country-level agreement between all databases is about 95.8% (1.57M addresses). The agreement between the databases might suggest more confidence in the geolocation results, it might also indicate a common incorrect source of the geolocation information (e.g., *registry data*).

We now turn to city-level resolution comparisons. Figure 3.4 shows pairwise comparison of databases locations (i.e., coordinates) for the Ark-topo-router addresses. For each pair of databases, we compute the distance between the locations from the two databases for each IP address. We then plot the distance distribution for all the addresses. Only the addresses with city-level and *(latitude, longitude)* coordinates in all databases are included (i.e., around 692k IP addresses). The pairwise comparison of the two MaxMind databases shows mostly small differences. 470k addresses (68%) have identical coordinates in the two databases and are truncated from their pairwise distance CDF. But for 11.4% of the addresses, the distance is more than 40 km, indicating that the IP address is likely geolocated to different cities. Other pairwise comparisons show more discrepancies where more than 29% of the addresses are geolocated

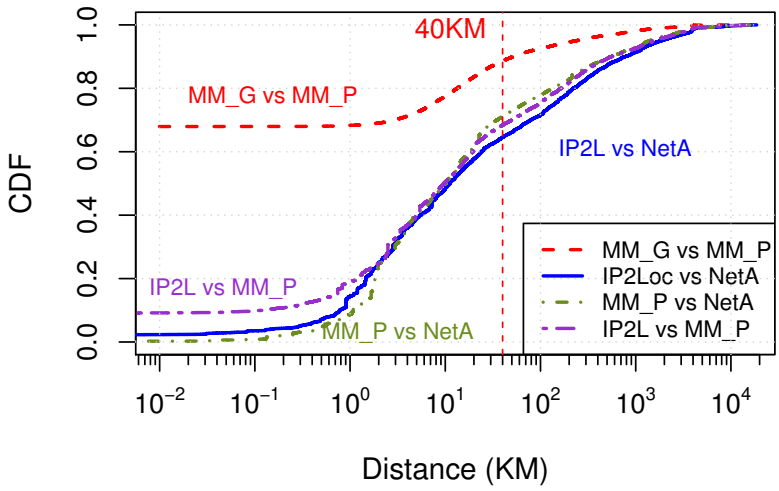


Figure 3.4: Databases pairwise distance distributions show at least 29% city-level disagreements for different vendors.

by different databases to locations more than 40 km apart. The CDFs for IP2Location-Lite and NetAcuity *vs.* MaxMind-GeoLite are omitted since they are similar to those *vs.* MaxMind-Paid.

3.5.3 Evaluation Using Ground Truth Data

Using our ground truth of 16,586 interface addresses (§3.3), we evaluate the coverage and accuracy of all databases at country- and city-level. The databases are accessed again on early July 2016, to geolocate the ground truth (i.e., about 50 days after creating the DNS-based set). We observe that 7.4% of our DNS-based set addresses likely moved during a 16 months period (§3.3.4.1). The movement is likely much less in 50 days (i.e., one-tenth of the 16 months) and is unlikely to affect our conclusions.

3.5.3.1 Databases' Coverage and Accuracy Over the Ground Truth

IP2Location-Lite and NetAcuity show near-perfect country- and city-level coverage for the IP addresses in the ground truth. MaxMind-GeoLite and MaxMind-Paid have around 95.4% country-level coverage, and only 30.4% and 41.3% city-level coverage, respectively.

The country-level geolocation accuracy is usually reported at higher than 97% by the geolocation service providers [113] (e.g., MaxMind GeoIP2 reports 99.8% accuracy [79]). However, our results over ground truth data show less accuracy for router geolocation. NetAcuity

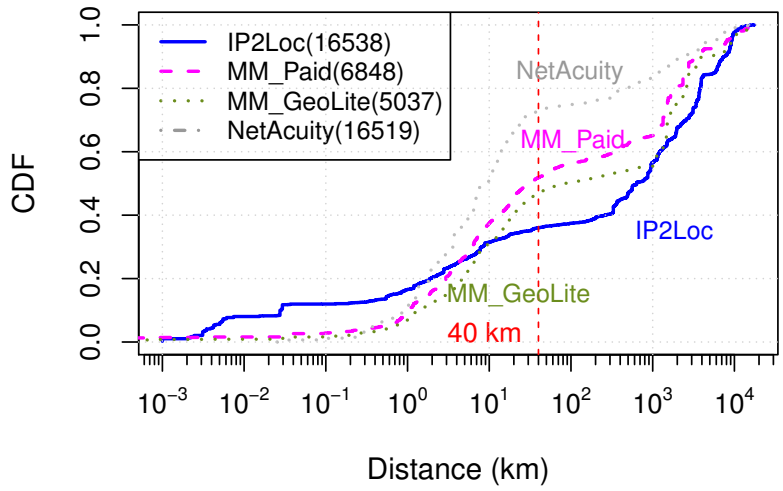


Figure 3.5: Databases *vs.* ground truth geolocation error. The number of addresses in each CDF is enclosed in parenthesis.

outperforms the other databases at only 89.4% accuracy while IP2Location-Lite and MaxMind databases are comparable with 77.5% to 78.6% accuracy. We discuss country-level accuracy in more depth when we break down results by RIR next in §3.5.3.2.

Figure 3.5 shows the distribution of the geolocation error for each database *vs.* the ground truth for the addresses with city-level geolocation. The vertical red line (at $x = 40$ km) is our city range threshold. NetAcuity has clearly better accuracy compared to other databases but still incorrectly geolocates some interfaces hundreds of kilometers away from their actual locations. IP2Location-Lite is the least accurate but has much better city-level coverage compared to both MaxMind databases.

3.5.3.2 Regional Evaluation

To study the accuracy of the databases regionally, we break down the ground truth addresses by their RIRs. Figure 3.6 shows the country-level accuracy by region. Each column in the graph shows the number of correctly and incorrectly geolocated addresses for each database. The percentage over each column shows the fraction of incorrectly geolocated addresses. From the graph we see that NetAcuity is the most accurate in all regions but there is still room to improve. We also observe that IP2Location-Lite and the two MaxMind databases' country-level accuracy results are comparable in all regions except for APNIC.

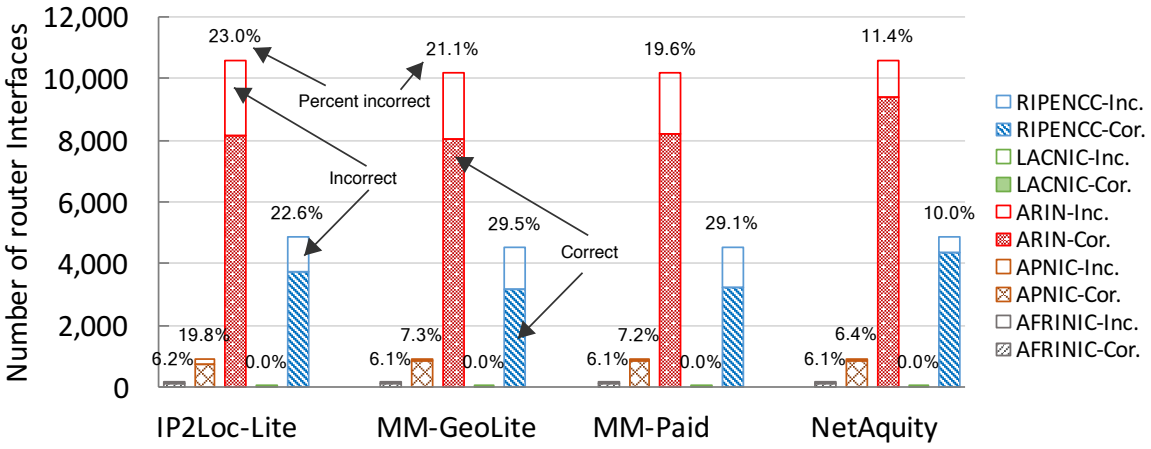


Figure 3.6: Country-level accuracy breakdown by RIR for ground truth. NetAcuity is the most accurate in all regions.

We go one step further and compare country-level accuracy for individual countries. Figure 3.7 shows the fraction of addresses correctly geolocated for the 20 countries with most addresses in the ground truth (country code and the number of addresses are depicted on the x-axis). While all databases show better than 94% accuracy for addresses in the United States (U.S.) and Russia, their accuracy in most other countries is relatively low, especially the IP2Location-Lite and the two MaxMind databases, which show surprisingly low accuracy in western countries, such as France and the Netherlands. IP2Location-Light, MaxMind-GeoLite, and MaxMind-Paid agree on the (incorrect) location of 2,277 addresses, which corresponds to around 61%, 64%, and 67% of their incorrectly geolocated addresses respectively. NetAcuity shows the most reliable results with at least 74% country-level accuracy in all 20 countries.

Finally, we evaluate city-level accuracy by region. Figures (3.8a, 3.8b, 3.8c, 3.8d) respectively show the distribution of geolocation error with breakdown by RIR for the IP2Location-Lite, MaxMind-GeoLite, MaxMind-Paid, and NetAcuity against the ground truth data. Only routers with city information in the databases are included (numbers are shown next to each RIR name). IP2Location-Lite has almost perfect city-level coverage but the accuracy is lacking, especially for ARIN addresses. Apart from ARIN, MaxMind seems to provide city-level geolocation only when it has some confidence in it, which could explain their low city-level coverage and relatively good city-level accuracy. For example, MaxMind-Paid city-level accuracy for the

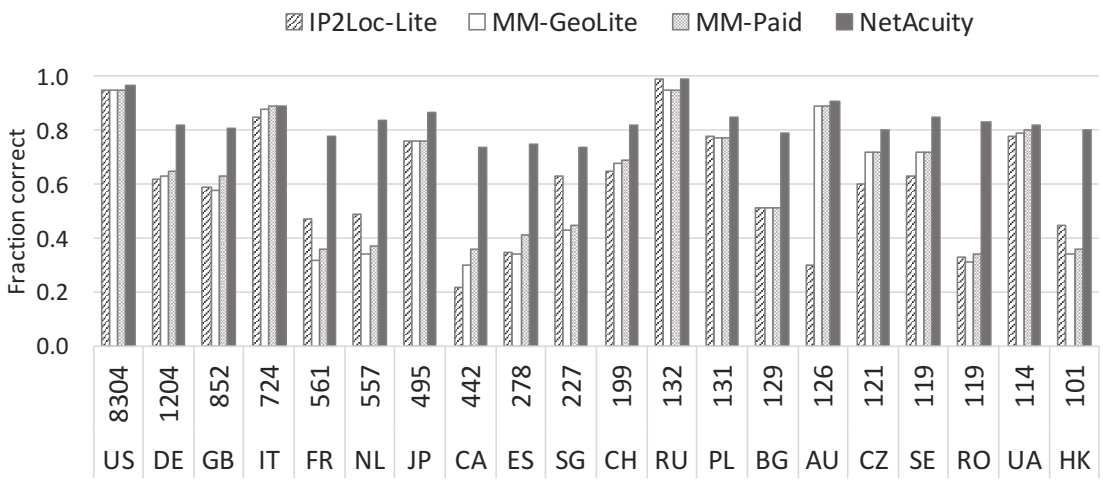


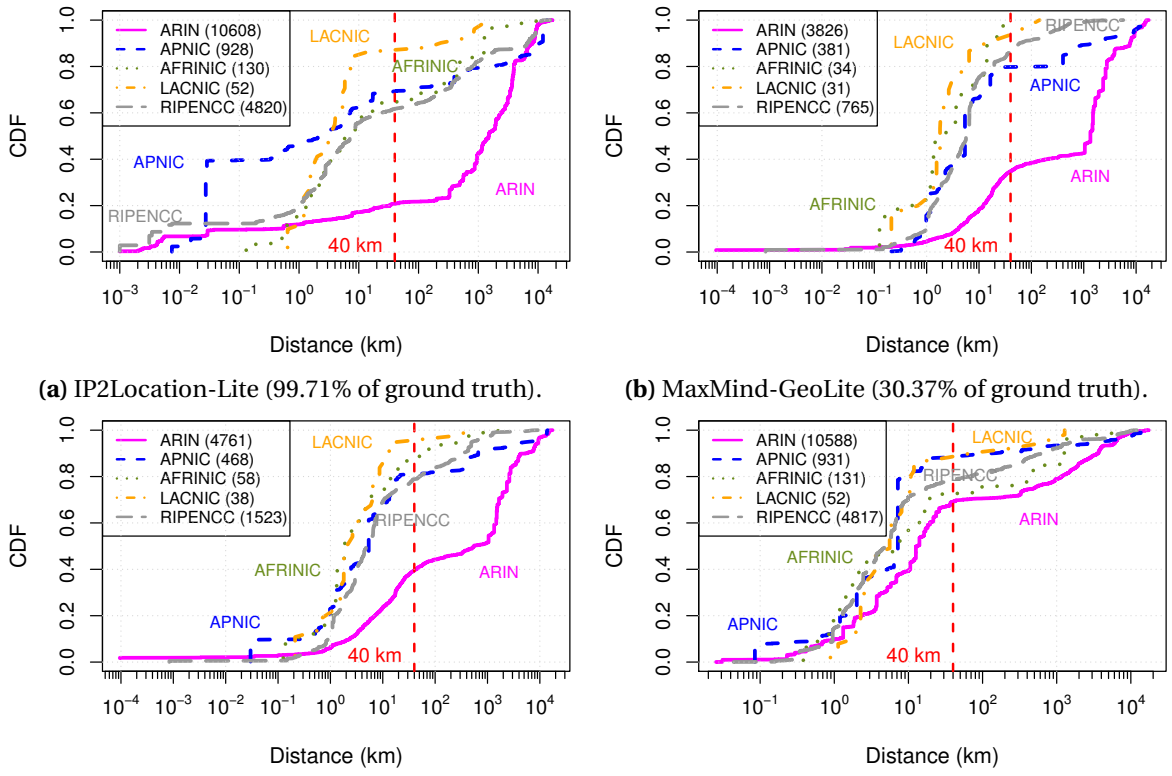
Figure 3.7: Databases' country-level accuracy is unreliable in most countries but NetAcuity is relatively consistent.

RIPENCC addresses is 78.9% with only 31.3% coverage compared to only 70.9% country-level accuracy and 93.3% coverage. NetAcuity, again, shows consistent coverage and accuracy results, but like the other databases, it is less reliable for the ARIN addresses.

3.5.3.3 Poor City-level Accuracy at ARIN

The worst city-level accuracy for all the databases is observed for ARIN addresses. We use MaxMind-Paid as a case study to understand reasons for such poor accuracy. ARIN has 10,608 addresses (64%) of the ground truth. 2,793 of those addresses are not located in the U.S.—according to the ground truth data. However, MaxMind-Paid, possibly relying on registry data, geolocates 1,955 of them (70%) to the U.S. We find that 519 (26.6% of the 1,955 addresses) have city-level geolocation in MaxMind, most of them (504 addresses) have disagreements greater than 1,000 km with the ground truth locations.

Total number of ground-truth addresses located in the U.S. is 8,304 (7,815 from ARIN and 489 from other RIRs). Total ARIN addresses located in the U.S. with city-level information is 3,897, of which 2,267 (58.2%) have geolocation error > 40 km—our city-range. About 91% of them have block-level—/24 block or larger—locations compared to about 78% of the correctly geolocated addresses at city-level. Block-level location assignments can be responsible for large



(a) IP2Location-Lite (99.71% of ground truth).

(b) MaxMind-GeoLite (30.37% of ground truth).

(c) MaxMind-Paid (41.29% of ground truth data).

(d) NetAcuity (99.6% of ground truth data).

Figure 3.8: Databases *vs.* ground truth geolocation error breakdown by RIR. Only routers with city information are included.

geolocation errors for interface addresses not co-located with the other addresses in their block. We do not investigate blocks co-locality in this work.

3.5.3.4 Databases *vs.* Separate Ground Truth Datasets

We evaluate the databases city-level accuracy against the DNS-based and the RTT-proximity datasets separately to find if they take advantage of the location hints in the hostnames of all the DNS-based dataset addresses. The RTT-proximity dataset has 1,335 addresses (27.6% of RTT-proximity data) that do not have DNS names. We do not know how many of the remaining RTT-proximity addresses have useful city-level location hints in their hostnames. Note that we include the 109 common addresses between the two datasets only as part of the DNS-based dataset. Overall, a database that uses DNS-based techniques to decode location hints in hostnames is expected to perform better on the DNS-based ground truth dataset.

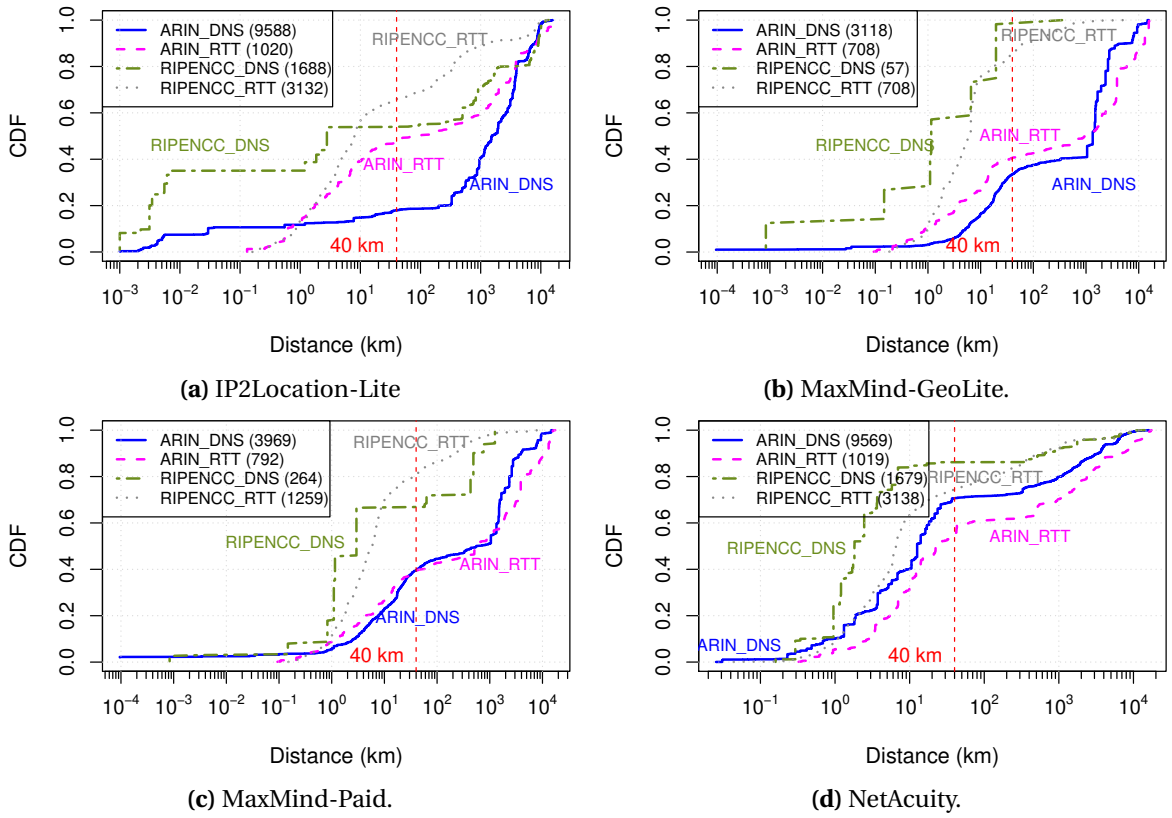
Figure 3.9 depicts the accuracy of the geolocation databases over the two sources of ground truth. NetAcuity is the only database that shows better city-level accuracy results over the DNS-based data compared to results over the RTT-proximity data. Considering our city-range threshold of 40 km, NetAcuity has 70.1% overall city-level accuracy over the RTT-proximity data and a better 74.2% accuracy over the DNS-based data. All other databases do worse over the DNS-based compared to the RTT-proximity data. MaxMind-Paid, for example, has only 43.9% overall accuracy over the DNS-based data and 66.5% over the RTT-proximity data. Regionally, NetAcuity shows better accuracy in all regions over the DNS-based data. For example, NetAcuity has 55.1% accuracy for ARIN addresses in the RTT-proximity data, and about 70.6% for ARIN addresses in the DNS-based data. According to these results, NetAcuity is the only database that might be using some DNS-based techniques to infer location hints from hostnames.

While the databases results over the RTT-proximity data look more competitive, NetAcuity still outperforms other databases over this dataset considering both accuracy and coverage. NetAcuity has a 70.1% city-level accuracy and 99.6% city-level coverage. The closest rival, MaxMind-Paid, has a comparable 66.5% accuracy but only 50.3% city-level coverage.

3.6 Recommendations

Based on our analysis of the geolocation databases using our datasets described in §3.2, we present our recommendations for using the databases with two thoughts in mind. First, our recommendations are mostly meaningful in ARIN and RIPENCC—where most of our ground truth IP addresses are located—and to a less degree in APNIC regions. Second, NetAcuity might have benefited from the nature of the DNS-based ground truth data (see §3.5.3.4), but we still argue that it has the best accuracy and city-level coverage as the results over both ground truth datasets show. With that in mind, here are the recommendations:

- If using a geolocation database is the only available option, we recommend NetAcuity to geolocate routers. Note that we think of NetAcuity city-level accuracy of 74.2% over the



(a) IP2Location-Lite

(b) MaxMind-GeoLite.

(c) MaxMind-Paid.

(d) NetAcuity.

Figure 3.9: Accuracy of the geolocation databases by dataset (DNS-based and RTT-proximity) for IP addresses in ARIN and RIPENCC. Only addresses with city information in the database are included (numbers in parenthesis). Only NetAcuity is consistently doing better over the DNS-based dataset in comparison to the RTT-proximity dataset.

DNS-based data as an upper bound for its overall accuracy. NetAcuity appears to benefit from the location hints encoded in the hostnames of the DNS-based dataset IP addresses.

- We do not recommend MaxMind databases if high city-level accuracy and coverage are required. The city-level accuracy is especially bad in the ARIN region. But we do see relatively good city-level results for MaxMind-Paid in RIPENCC and APNIC regions. However, the city-level coverage is very low.

- The commercial version of MaxMind is recommended over the public version if city-level accuracy and better coverage are required.

- We do not recommend IP2Location-Lite, the overall accuracy is too low especially at city-level.

- If the price is a problem and an overall 78% country-level accuracy is acceptable, the IP2Location-Lite and both versions of MaxMind were comparable. That said, the accuracy can be very low for some countries (see Figure 3.7).

- We recommend users not to trust city-level accuracy in ARIN regardless of the database used. NetAcuity was the most accurate there, but only 66% of the ground truth interface addresses there are geolocated to within 40 km of their actual locations.

3.7 Conclusions

This chapter evaluated router geolocation in four widely used geolocation databases. We examined the consistency and coverage of the databases using a dataset of 1.64M router interface addresses.We showed that the databases generally agree on the country-level (95.8% of the time), but the databases—from different vendors— show more discrepancy at city-level with more than 29% pairwise disagreements.

The results presented in §3.5 motivate our work, highlighted in the thesis statement (§1.2), to enable continuous, more reliable geolocation (Chapter 4 and Chapter 5). We evaluated the accuracy of the studied databases with a ground truth dataset of 16.6k IP addresses of router interfaces we geolocated using DNS-based and latency-based methods. The results showed that the databases can be inaccurate at geolocating routers at both country-level (only 89.4% or less of ground-truth addresses were correctly geolocated) and city-level (only 73% or less of the ground-truth addresses were correctly geolocated). Furthermore, the two MaxMind databases showed low city-level coverage at 43% for the MaxMind-GeoLite and 61% for the MaxMind-Paid.

Using the domain-specific ground-truth heuristics (§3.3.1), we showed that 7.4% of all DNS-based addresses appear to have moved over 16 months.This result supports our claim that we need continuous geolocation to keep up-to-date location mappings (Chapter 5).

This study also presented a breakdown analysis by RIR, showing the databases are less reliable at the city-level resolution at ARIN compared to other regions. NetAcuity shows the best

combination of coverage and accuracy. MaxMind shows relatively good city-level accuracy in regions other than ARIN, but it lacks extensive city-level coverage. Overall, these results show that researchers need to understand the impact of databases' inaccuracy on their results and pay extra caution when using them.

Chapter 4

IP Blocks Co-locality

Our thesis is about enabling state-of-the-art geolocation approaches to scale up as continuous geolocation services to the entire Internet to achieve more reliable, up-to-date geolocation. Geolocating IP addresses as groups of adjacent IP addresses (e.g., a /24 block) is a common approach to improve efficiency and enhance the coverage of a geolocation system [62,95]. Many services assume that the addresses within the same /24 prefix (a */24 block*) are geographically proximate—the *block co-locality assumption*. When this assumption is violated, some addresses in the block will have poor geolocation accuracy. Moreover, geolocating IP addresses individually or as small groups such as a /24 block could waste chances of more efficient geolocation for larger groups of co-located addresses (e.g., addresses in a /16 block used within a campus of an academic institution).

In this chapter, we present a delay-based method to assess the co-locality (geographic proximity) of adjacent IP addresses (§4.3). Our results support our thesis statement, showing that we can identify clusters of co-located addresses that we can treat as units to enable efficient geolocation (§4.6) while avoiding geolocation error caused by incorrect co-locality assumptions (§4.5).

We first develop a hierarchical clustering method to cluster IP addresses by the similarity of their Round Trip Times (RTTs) observed from several vantage points (§4.3). We validate this method with ground truth data in §4.4.3. We use our method to evaluate the block co-locality assumption over a large dataset of 1.41M /24 Internet blocks (§4.5). We then show we can use our method to identify arbitrary-size clusters of co-located IP addresses (§4.6).

4.1 Introduction

Achieving good IP geolocation accuracy can be difficult when some blocks of adjacent IP addresses span large geographic areas [43,115]. The depletion of IPv4 address pools may play part

in more /24 blocks spanning larger geographic areas [28,104]. On the other hand, identification of large groups of co-located IP addresses allows for a more efficient, scalable geolocation.

The block co-locality assumption is common among geolocation service providers. For example, public geolocation databases such as MaxMind *GeoLiteCity* [81] and IP2Location *LITE-DB11* [68] adhere to the block co-locality. The entries in these databases identify IP blocks of various sizes and assign each a specific location. The public IP2Location database has 2.67M entries covering the entire IPv4 address space (no locations are assigned to special blocks such as multicast [123]). In these databases, nearly all blocks are /24 or larger; thus, 99% or more of the /24 blocks are marked as co-located.

Some location-dependent applications also seem to adhere to the /24 co-locality assumption. An architecture proposed by Chen *et al.* [22] maps a client's request to a proximal content server based on prefixes, meaning that all clients within the same prefix are mapped to the same content server. They suggest the mapping at /20 prefix granularity to minimize the number of required mappings.

In this chapter, we present a delay-based clustering method to identify co-located IP addresses (i.e., geographically proximate addresses). We use our method to assess the /24 block co-locality assumption. We also show how our method can help a geolocation system scale-up by identifying larger groups of co-located IP addresses to geolocate them as a unit.

We assess IP addresses co-locality based on the observation that geographically co-located, adjacent hosts will show similar network delays when probed by the same vantage points [91]. Fan *et al.* used a similar clustering technique to find Front End (FE) servers that belong to one CDN [42].

To evaluate our method, we leverage a large subset of the dataset collected by Hu *et al.* [61] and publicly available [96]. The dataset contains round-trip estimates for every responsive address in the IPv4 address space measured from several vantage points (VPs).

To assess the /24 block co-locality assumption, we cluster the IP addresses in each /24 block into groups by the similarity of their delay observations from multiple VPs. We then identify

/24 blocks with multiple clusters and show that these clusters violate the block co-locality assumption and likely contain addresses in distinct locations. We also apply our method to IP addresses within /16 blocks that we believe are constrained in a small geographic area to evaluate our method on larger groups of co-located IP addresses.

Our first contribution is to introduce and evaluate a methodology to assess the co-locality of endpoints in an IP block. We first use the method to assess the /24 blocks co-locality assumption. Our delay-based clustering algorithm automatically identifies blocks that appear to have endpoints at different locations. We validate the accuracy of this method against two ground truth datasets (§4.4): first, a set of /24 blocks we selected based on our belief that they are co-located, and second, an artificially-constructed set of multi-location blocks. We confirm that 93% of the blocks identified as multi-location blocks in the ground truth datasets are true positives. Our second contribution is the application of this methodology to analyze 1.41M /24 blocks (118M addresses). We find that a noticeable fraction of these blocks (17%, or 247k blocks) appear to have endpoints at multiple locations with an upper bound false positives rate of 5.4%. Finally, we show that the clustering method can identify large groups of adjacent, co-located addresses. We evaluate our method over 65 different university groups of IP addresses that we believe are co-located. Our method correctly identifies the majority of addresses in each university block as co-located.

4.2 Dataset Description

Our analysis uses the latency estimates from the IP geolocation dataset collected by researchers at ISI [96] extended from prior work by Hu *et al.* [61]. The original dataset contains round-trip time measurements for all the allocated and responsive IP addresses in the IPv4 address space. The dataset has about 472M IP addresses in just under 3.5M /24 blocks and was collected from Feb. 2012 to Mar. 2013. The RTTs are measured from about 670 vantage points (VPs) on PlanetLab [24]. The work used the following algorithm to pick the 10 closest VPs to any

/24 block. First, all available VPs probe a few representative IP addresses in a block. The 10 VPs with shortest RTTs are then selected to probe all IP addresses in that block.

The use of VPs close to the target minimizes interference from congestion and maximizes the precision of geolocation (something 400 milliseconds away can be anywhere on earth, but something within few milliseconds is likely in the same city). Also, to reduce congestion noise, latency was reported as the *minimum* of 10 measurements. For our work, we use the raw probing data for all /24 blocks where each block contained at least 10 IP addresses that responded to *all* VPs probes. The delay measurements of each IP address are treated as its coordinates in a multidimensional space. Our dataset comprises of **118.5M** IP addresses in **1.41M** /24 blocks.

4.3 Identification of Co-located IP Addresses

We develop our methodology to identify co-located IP addresses based on the insight that geographically co-located IP addresses from the same IP block exhibit relatively similar network delays when probed from the same vantage points [91]. Unlike most delay-based geolocation methods, our method does not map *VP-to-target* observed delays into distance constraints on the target location (§2.3.3). Instead, we use the observed delay as signatures that identify if the addresses in an IP block are proximate. We describe our methodology in §4.3.1 and its limitations in §4.3.2.

4.3.1 Methodology

In order to identify co-located IP addresses, we formulate the problem as finding similar IP addresses in a multidimensional space of delay coordinates. For each IP address we create a vector of 10 delay measurements observed from 10 different VPs (§4.2). The use of multiple VPs has two advantages: (a) provide some protection against noise in the measurements, (b) provide better delay signatures for the clustering method to identify similar objects.

Co-located IP addresses are expected to have small distances between them in the delay multidimensional space. So we cluster IP addresses in a block based on the similarity of their

delay vectors. A block with all of its IP addresses mapped into one cluster is likely a single-location block, while a block with 2 or more clusters is likely a multi-location block.

To cluster the IP addresses based on the similarity/dissimilarity of their delay vectors, we use an agglomerative hierarchical clustering algorithm from the **R** *cluster* package called *agnes*. Given the delay vectors, the algorithm generates a tree-like hierarchical structure, *dendrogram*, based on the dissimilarities of the delay vectors. We use the *Standardized Euclidean* distance metric to measure the dissimilarities. We use a *dynamic tree cut* method from the *dynamicTreeCut* package [74] to identify the clusters in the dendrogram. The combination of these methods satisfies the need to identify clusters automatically without prior knowledge of their number or size.

As with other agglomerating hierarchical methods, the *agnes* method generates a bottom-up hierarchical structure for the input observations. Each observation starts as a cluster by itself. In each subsequent step, the closest two clusters not already in the same cluster are merged into one larger cluster. The process continues until there is only one cluster of all observations. The height at which two clusters are merged in the tree-like dendrogram is computed as a function of the dissimilarity between the two merged clusters. The dissimilarity between two clusters can be computed in different ways. In this work, we use the *average linkage* method, which computes the distance between two clusters as the average of pairwise dissimilarities between the objects in the two clusters. The main advantage of using average linkage is to alleviate the effects of outliers in the delay measurements. For two clusters, say cluster A with n_a objects and cluster B with n_b objects, the metric is computed using Equation 4.1, where D is the distance metric used to compute the distance between two objects. We use the *Standardized Euclidean* distance metric to balance the depth of the measurements observed from VPs at different distances from targets.

$$d_{average}(A,B) = \frac{1}{n_a n_b} \sum_{i=1}^{n_a} \sum_{j=1}^{n_b} D(IP_{Ai}, IP_{Bj}) \tag{4.1}$$

A standard method to identify clusters in a dendrogram is to cut it at a *fixed height* based on expected results in a given application. This method can work well for many applications, but in our case, prior work has shown the need for selecting clustering thresholds dynamically when examining Internet RTT data [42]. To identify clusters automatically for each of our 1.41M /24 blocks, we use the "*Dynamic Hybrid*" tree cut method [74] to identify clusters in a dendrogram dynamically. This method uses dendrogram-merging information to build the clusters in a bottom-up fashion. Many of the method's parameters are set as a fraction of the joining heights of the dendrogram branches. The one parameter we found most effective is the *minimum gap* parameter, which specifies the minimum joining height to allow two clusters to be merged. Higher settings of this parameter allow more smaller clusters to be merged. The result is fewer clusters with significant dissimilarities indicating higher probability of being at different locations. We also set the minimum cluster size to 10 to reduce the possibility of getting small clusters of outliers.

4.3.2 Methodology Limitations

Our delay-based clustering method has the following limitations. First, as with all delay-based methods, our approach can be affected by measurement inaccuracy caused by transient network events such as congestion. This problem is alleviated by taking multiple measurements over time, the use of multiple VPs per block and picking the minimum RTT. In the case of lingering network events, such as a persistent congestion or a routing change, we expect the measurement of adjacent co-located addresses to be affected comparably. As a result, our method will still be able to identify these addresses as co-located. Second, our methodology does not identify the geographic locations of the clusters and the actual distance between them. Fortunately, geographical locations are not required to determine if two IP addresses are *co*-located as we propose in our method, just as one can determine the heavier of two objects without knowing their actual weights.

4.4 Validating Identification of Multi-Location Blocks

In this section, we validate our method by showing that it accurately finds single- and multi-location blocks in our ground-truth dataset. We build our ground truth dataset as follows. First, we identify single-location /24 blocks as described in §4.4.1. Second, we use this data to construct a multi-location ground truth dataset in §4.4.2. Third, we use the constructed ground truth dataset to validate our methodology in §4.4.3. Finally, we estimate an upper bound of false positives for the clustering method as described in §4.4.4.

4.4.1 Building a Single-Location Ground Truth Dataset

We build a dataset of /24 blocks that we strongly believe are single-location blocks for two purposes: (a) to evaluate the clustering method accuracy on single-location blocks, (b) to build the multi-location ground truth dataset. Our single-location ground truth dataset is composed of address blocks belonging to selected academic institutions. We opted for academic institutions because they typically have specific, distinct physical locations with many end-user computers within a small geographical area. Such institutions often host their own web services [128]. Academic institutions are also relatively easy to find on the map, and services such as Google Maps already have campus outlines and geographic coordinates for them. Finally, academic institutions tend to be long-lived with more accurate *WHOIS* entries than average. These properties make academic blocks attractive candidates for our purposes.

We begin by identifying the locations and /24 IP address blocks containing the websites of 4,650 universities from different locations around the world listed at [2]. We verify that these blocks are locally hosted at their universities by verifying them against outsourcing. We detect outsourcing using *WHOIS* information and filter out outsourced blocks. We looked up the organization name from *WHOIS* databases and matched it with the institution name to identify outsourcing. For example, Duke University's website—at the time of publishing this work— was at the IP address (54.191.241.8), which the *WHOIS OrgName* identifies as *Amazon Technologies Inc.*, this shows evidence of outsourcing and is therefore removed from our list.

We cross-checked the remaining blocks from the university dataset with the one we extracted from the 1.41M blocks (§4.2). We found 560 /24 blocks with valid data (10 IP addresses or more with measurement from 10 VPs). We used the *Google Maps Geocoding API* [1] to identify a university's physical location (latitude/longitude).

We apply one more filter to satisfy our goal of building a ground truth of synthetic multi-location blocks from the single-location blocks (§4.4.2). Since we cannot combine blocks probed by different sets of VPs (§4.4.2), we only include a /24 block in our single-location dataset if at least one other single-location block is probed by the same set of VPs (termed *VP-compatible blocks*). (We relax this restriction in §4.4.4 to experiment with a larger set of single-location /24 blocks.) We found 85 such /24 blocks, which we use as our single-location dataset and to build the multi-location ground truth dataset as described next in §4.4.2. The filters described above are rigorous and resulted in rejecting otherwise viable entries in our dataset. However, this only increased the confidence of the remaining entries since they passed a higher bar.

4.4.2 Building a Multi-Location Ground-Truth Dataset

We built a multi-location ground truth dataset by combining two single-location blocks to form synthetic multi-location blocks as Table 4.1 shows. We used only 77 out of the 85 single-location blocks to form the multi-location blocks (§4.4.3). To generate synthetic multi-location blocks, we find all blocks from the single-location dataset that were probed by the same set of VPs; we call these *VP-compatible blocks*. We then computed all two-block combinations in each set of VP-compatible blocks, combining all measurement data from the two blocks to create a new synthetic block. Merged blocks may have up to 512 addresses; however, since we had data from ping-responsive addresses only, merged blocks almost always had fewer addresses, often less than 256. These synthetic blocks form our ground truth multi-location blocks dataset.

Some of the VP-compatible single-location blocks that we use to build the multi-location dataset are quite close to each other. We, therefore, identify two subsets in our multi-location dataset: those composed of *almost-co-located* blocks (within 13 km of each other), and *not-co-*

Table 4.1: Building the synthetic multi-location dataset from single-location blocks.

single-location blocks	85
used to build multi-location blocks	77
synthetic multi-location blocks	120
not-co-located	99
almost-co-located	21

located blocks (35 km apart or more[2]). We identify 21 almost-co-located synthetic blocks and 99 not-co-located blocks.

4.4.3 Validation

The two ground truth datasets (single- and multi-location) let us evaluate the ability of our delay-based clustering method to identify co-located blocks and blocks that span multiple geographic locations. We next describe the results of applying our method on both single- and multi-location datasets.

We first considered our single-location dataset. Our clustering algorithm classified correctly 91% (77 of 85) of the blocks in our single-location dataset. Seven blocks were identified to have 2 clusters while one block was not clustered. Manual investigation of the 7 blocks with 2 clusters showed distinct latency distributions for the two clusters. Hostnames and *traceroute* results did not show any evidence that the IP addresses in any of these blocks were at different locations. We still believe that these blocks are co-located, but some addresses experienced different delays, possibly due to wireless connections, where wireless access points are known to add a few milliseconds to the hosts' observed RTTs [90].

We next turn to our synthetic, multi-location dataset. First, we discarded synthetic blocks built from the 7 misclassified single-location blocks since we know those would be identified as multi-location blocks. We then applied our clustering methodology to the remaining not-co-located 99 synthetic blocks. Figure 4.1 shows the number of identified clusters and the corresponding distance between the two combined blocks for each synthetic block. We cor-

[2]The single-location dataset does not have VP-compatible blocks within 13 km to 35 km from each other.

rectly identified 88% of these as multi-location blocks. Manually investigating the remaining 12% false negatives, we saw very similar delay measurements for IP addresses in the combined blocks leading to incorrect identification. Such delay measurements could be a result of a relatively small distance between the combined blocks Other possible reasons are blocks sharing most of the network hops to the VPs or similar path distances from the VPs. For example, the VP at the Berlin Institute of Technology observed similar delay measurements to the two synthetic blocks at the University of Göttingen and Jade University of Applied Sciences in Germany. The VP at Hamburg University of Applied Sciences also observed similar delay measurements to these two blocks. The clustering method falsely identified these two blocks as co-located.

To examine the most challenging synthetic blocks, we also looked at the 21 proximal almost-co-located synthetic blocks, where the real-world distance between each block is within 13 km. One example is the combination of Dongbei University of Finance and Economics and the Dalian University of Technology in China, which are about 2 km away from each other. Despite close physical proximity, our method correctly identifies 38% (8 of 21) of these blocks as multi-location. Overall, between the single- and multi-location datasets, 93% of the cases identified as multi-cluster blocks are true positives, which gave us confidence that our methodology works reasonably well.

4.4.4 Bounding the False Positives

Our clustering method needs to maintain a low false-positive rate to ensure that we do not overestimate the number of multi-location blocks (false positives). To estimate an upper bound for the false positive rate, we built another extended set of /24 blocks that are likely co-located. We leveraged again address blocks in academic institutions. We followed a similar procedure as with the single-location dataset, but now we picked a set of 100 random universities from the set of 560 academic institutions we have valid data for (§4.4.1) and that have at least a /16 block assigned to them and verified not to include web hosting services. We found 3,062 /24 blocks within the selected 100 /16 blocks that appear in the ISI dataset. Table 4.2 shows the results of

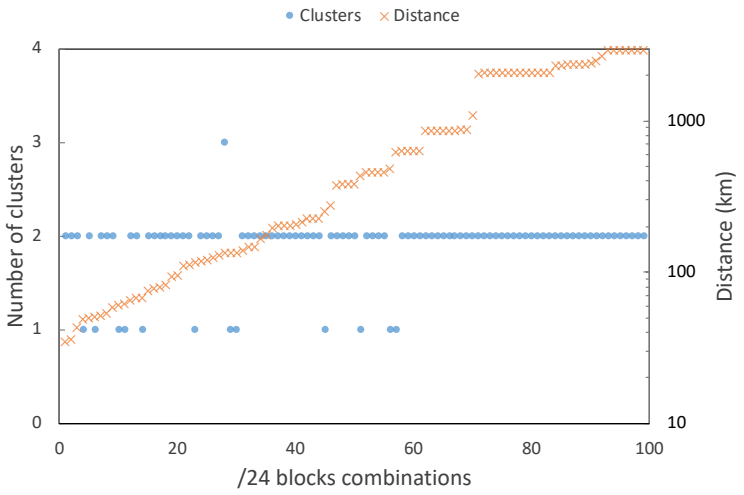


Figure 4.1: Results of applying the delay-based clustering to 99 2-block combinations. The graph shows the number of reported clusters for each synthetic block and the corresponding distance between the combined blocks.

Table 4.2: Results of clustering university /24 address block to identify a false positives upper bound.

number of institutions	100	
number of /24 blocks with valid data	3,062	100%
one cluster	2,657	86.8%
two clusters	166	**5.4%**
not clustered	239	7.8%

running the clustering method on all of these blocks. The results show that 239 blocks (7.8%) are not clustered since they did not meet our criteria for the minimum number of addresses to form a cluster, 2,657 blocks (86.77%) have one cluster, and 166 blocks (5.4%) have 2 clusters. Since any of the blocks identified as multi-location could indeed be multi-location blocks, we regard the 5.4% as an approximate upper-bound for the false positive rate in our clustering method.

4.5 Co-Locality of /24 Blocks in the Wild

In this section, we present the results of applying our clustering method to 1.41M /24 blocks in the ISI dataset (§4.5.1). We then present a characterization of the blocks identified as multi-location (§4.5.2).

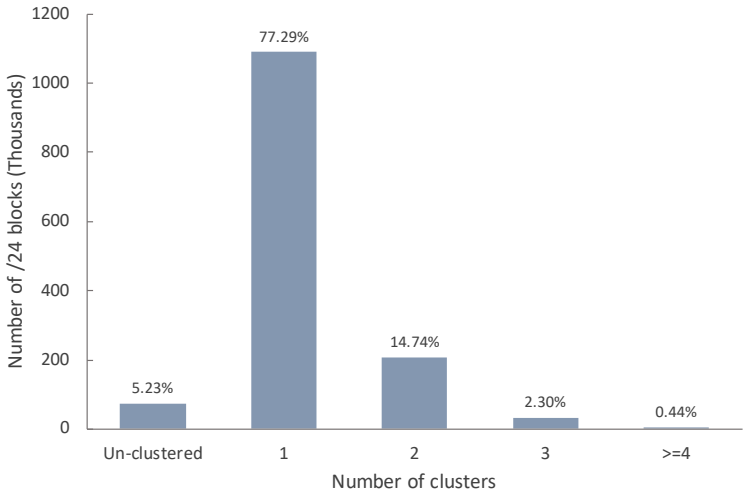


Figure 4.2: Distribution of the number of clusters for all 1.41M /24 blocks in the ISI dataset. About 17% of the /24 blocks (~247k) are identified as multi-location.

4.5.1 Identifying Multi-Location /24 Blocks

Figure 4.2 shows the distribution of identified clusters in all 1.41M /24 blocks. About 17% (~247k blocks) appear to have endpoints at multiple locations. 82% of the multi-cluster blocks are grouped into 2 clusters of IP addresses. A small fraction, 0.44%, of the multi-location blocks are grouped into four or more clusters. Our method failed to cluster 73,792 /24 blocks (5.23% of all blocks), 98% of which have 20 IP addresses or less. A block is identified as not clustered when the clustering method can not find any cluster with the required minimum number of IP addresses that satisfies all clustering criteria. While this is more typical in blocks with a small number of IP addresses, it can also be true for any block with endpoints that are highly scattered geographically.

4.5.2 Characterizing Multi-Location /24 Blocks

Our method identified about 247k blocks as multi-location blocks. We found multi-location blocks in 182 different countries. Table 4.3 lists the top 10 countries sorted by the number of /24 blocks (second column) found in our 1.41M blocks dataset. The third and fourth columns respectively list multi-location /24 blocks identified per country both as an absolute number and as a percentage of a country total /24 blocks in the dataset. We note different percentages of

Table 4.3: Top 10 countries sorted on their total number of /24 blocks in the dataset and the corresponding multi-location blocks percentage per country.

Country	Blocks in Dataset	Multi-location Blocks	Multi-location Blocks %	% of Country Blocks in Dataset
US	430947	84649	19.64%	6.83%
CN	98016	1507	1.54%	7.45%
DE	81925	34691	42.34%	17.45%
JP	71131	20899	29.38%	8.97%
GB	63609	12339	19.40%	13.24%
KR	60265	10296	17.08%	13.73%
FR	55870	6900	12.35%	17.97%
BR	53050	4772	9.00%	16.57%
RU	37772	2954	7.82%	21.10%
CA	35816	3414	9.53%	12.57%

identified multi-location blocks across different countries. For example, Germany has 42.34% of its blocks in the dataset identified as multi-location, while only 1.54% of the /24 blocks in China are labeled as multi-location. The differences in multi-location percentages across countries may be due to different policies of IP address assignment. We find that a significant portion of the identified multi-location blocks belong to big ISPs in countries with rich Internet infrastructure such as the United States and Western Europe. (See the discussion of Table 4.4 next.) The fifth column lists each value in column 2 as a percentage of the total blocks assigned to each country by the corresponding RIR. From the table, we can see our dataset has a reasonable representation of the IPv4 address space for the top 10 countries, ranging from about 7% to over 21%.

In Table 4.4, we list the top 10 ISPs and their Autonomous System Numbers (ASNs) sorted by the number of /24 blocks (third column). The fourth column lists the total number of blocks for each ISP in the dataset, while the fifth column is simply the ratio of the previous two columns. From the table, we can see that some ISPs (such as DTAG and ONC NTT) show a high percentage of multi-location blocks, while others have much smaller percentages. This difference reflects different IP assignment policies across ISPs concerning the geographic distribution of the addresses. It is also worth mentioning that some ISPs dominate the multi-location blocks

Table 4.4: Top 10 ISPs sorted on their number of multi-location blocks, and their corresponding percentages of ISPs' total number of blocks in the dataset.

ISP Name	ASN	Multi-loc. Blocks	Blocks in Dataset	Multi-loc. Blocks %	% ISP Blocks in Dataset	Country
DTAG Deutsche Telekom AG	3320	21,204	36,359	58.3%	25.5%	DE
COMCAST-7922 - Cable Comm	7922	11,804	69,117	17.1%	24.2%	US
OCN NTT Comm Corp.	4713	9,204	14,841	62.0%	12.9%	JP
ATT-INTERNET4 - AT&T Serv	7018	8,994	43,656	20.6%	11.8%	US
Uninet S.A. de C.V.	8151	7,033	24,269	29.0%	49.0%	MX
UUNET - MCI Comm Services	701	6,881	28,497	24.2%	14.7%	US
CENTURYLINK-US-LEGACY-QWEST	209	6,766	26,197	25.8%	38.8%	US
BSKYB-BROADBAND AS Sky UK Ltd	5607	5,810	10,501	55.3%	40.7%	GB
VODANET Vodafone GmbH	3209	5,665	14,049	40.3%	41.5%	DE
TPNET Orange Polska Spolka	5617	5,561	14,950	37.2%	46.6%	PL

in their countries in our dataset. For all the countries listed in Table 4.4, more than 40% of their multi-location blocks are from few (less than four) ISPs. We leave further investigation of these phenomena as future work. The sixth column lists each ISP blocks in the dataset (column 4) as a percentage of the total blocks in its ASN announced prefixes (computed based on data from `http://cyclops.cs.ucla.edu/`). From column six, we can see that our dataset has a reasonable representation for the top 10 ISPs ranging from about 12% to 49%.

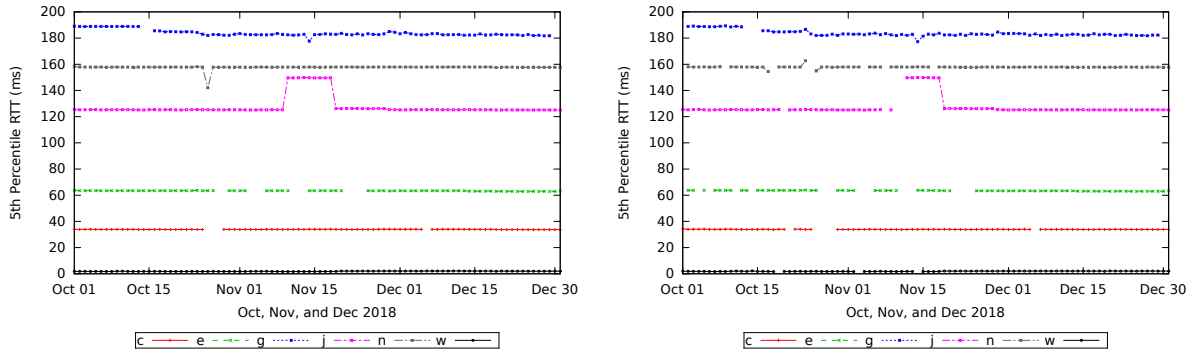
4.6 Identifying Arbitrary-Size Clusters of Co-located Addresses

In this section, we show that our delay-based clustering method can be used as an optimization tool for efficient geolocation. We already examined our method over /24 blocks (§4.4.3), showing that we can identify groups of adjacent, co-located IP addresses within a /24 block. Next, we show that our method can also identify clusters of arbitrary-size clusters of co-located IP addresses. We first demonstrate the similarity of latency observations of co-located blocks in §4.6.1. We then build and describe our evaluation dataset in §4.6.2. Finally, we show the result of applying the delay-based clustering to the evaluation dataset in §4.6.3.

4.6.1 Similarity of Co-located Blocks Latency

To demonstrate the similarity of RTT-measurement of co-located /24 blocks from the same organization, we use data from the *USC ping dataset* (§5.2.1). Figure 4.3 depicts RTT data extracted for two sample /24 blocks from the 128.125/16 block at the University of Southern California (USC). Each line, in either of the graphs, represents the data from one of the 6 vantage points used to probe a block over the last quarter of 2018 (2018q4). Each data point on any line represents the 5%ile of one day worth of RTT observations (close to minimum RTT approximation), where each VP probe a random IP address in the block around 130 times a day with 11 minutes between the measurements.

From the graph, we can see the two blocks have near-identical RTT fingerprints in any day during 2018q4. Indeed, except for a few less responsive blocks, we can generalize our observation about the two blocks in Figure 4.3 to 107 other /24 blocks from USC for which we have latency data. These observations are also valid for several other academic institutions that we investigated. From these observations, we expect our delay-based clustering method to be able to identify arbitrary-size clusters of co-located IP addresses from one organization.



(a) 128.125.31.0/24 **(b)** 128.125.49.0/24

Figure 4.3: Two /24 address blocks with near-identical RTT fingerprints from USC /16 prefix.

4.6.2 Evaluation Dataset

To examine the clustering method over larger groups of IP addresses known to be in one location, we again use the blocks from the dataset of 100 universities from §4.4.4. This time, rather than looking at individual /24 blocks, we look to identify any number of co-located IP addresses in a university /16 block for which we have valid data. We assume a university is likely to assign its allocated address range to end-user computers within a defined small geographical area that the university owns. Using information from the websites of the universities, we confirmed that the data we are using in this evaluation belongs to universities that have one campus or a few campuses within the same city.

Since our method clusters IP addresses by the similarity of their latency observations, we can only apply the method to *VP-compatible* blocks (i.e., blocks probed by the same set of VPs). Similar to building the synthetic dataset in §4.4.2, we find all *single-location* blocks from §4.4.4 that are *VP-compatible* in each university prefix. We require a university prefix to have at least two VP-compatible /24 blocks to be included in our evaluation. Often, we find that /24 blocks of one university are probed by identical or near-identical sets of VPs. For this evaluation, we picked the largest group of /24 blocks probed by an identical set of VPs for each university. We find 65 universities with a total of 1,974 /24 blocks that satisfy our criteria. We extract the RTT-vectors of the IP addresses in each university group, as described in §4.3.1, and use these vectors as the input for the delay-based clustering method next.

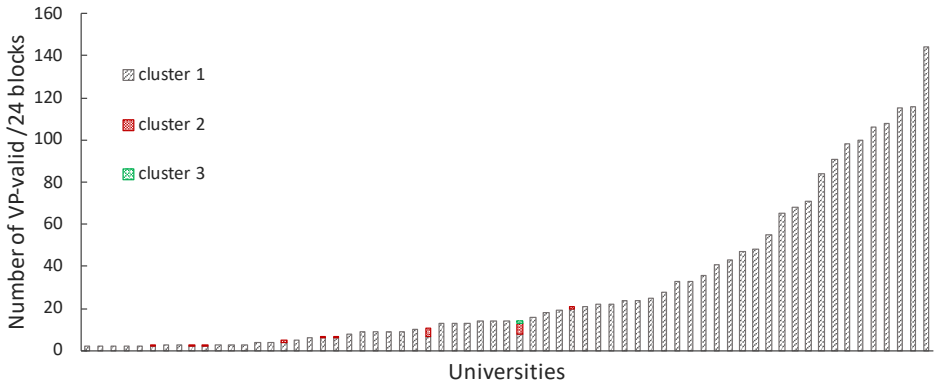


Figure 4.4: Results of clustering 65 university groups of IP addresses in VP-compatible blocks.

4.6.3 Results

In this section, we examine the clustering method over 65 groups of co-located IP addresses. We form each group from addresses in two or more VP-compatible /24 blocks (§4.6.2). Given that the addresses in one group are likely co-located, we expect our method to identify the addresses in each group as one cluster.

Although our method is not aware which RTT-vectors (of IP addresses) belong to a given block, almost all RTT-vectors of each /24 block ended up in the same cluster. This result suggests homogeneous use of the IP addresses in those blocks. Figure 4.4 shows the clustering results for each of the 65 university groups sorted by the number of VP-compatible blocks a university has (y-axis), while Table 4.5 presents the overall results.

Our method correctly identifies the IP addresses of 56 (86%) of the 65 groups as one cluster each (columns with dark stripes in Figure 4.4). We identify the addresses in 8 groups in 2 clusters and one group in 3 clusters, indicating endpoints at different locations. We hypothesize some of these university blocks are connected via WiFi access points. We show in Chapter 6 that IP addresses in WiFi blocks can exhibit distinct latency patterns compared to those in fixed-line blocks. These distinct latency patterns could lead our method to identify such blocks as not-co-located even when they are nearby. We did not investigate these blocks further. Overall, 1,957 (99.1%) of the /24 blocks in the evaluation dataset are classified in the majority cluster of each

Table 4.5: Overall results of clustering IP addresses in the university VP-compatible blocks dataset.

institutions that satisfy selection criteria	65	100%
one cluster	56	86%
two or three clusters	9	14%
all institutions /24 blocks with valid data	1,974	100%
in the majority cluster	1,957	**99.1%**
not in the majority cluster	17	0.9%

university. These results show that our method can identify arbitrary-size groups of co-located IP addresses with reasonable accuracy. Moreover, these results support our thesis statement by showing that we can identify groups of co-located addresses by their latency observations, allowing us to geolocate them efficiently as units.

4.7 Conclusions

We presented a delay-based methodology to assess the co-locality of Internet addresses. Our method identifies whether or not a set of adjacent addresses are in one location based on the similarity of their latency estimates observed from several vantage points.

This chapter provides evidence to support part of our thesis statement (§1.2). Our thesis states that we can identify clusters of co-located IP addresses from latency observations to enable efficient geolocation. Using our delay-based clustering method, we showed that we can identify arbitrary-size clusters of co-located IP addresses—addresses that we can geolocate as a unit rather than individually—, allowing more efficient geolocation that avoids incorrect co-locality assumptions. We evaluated our method over IP addresses in 1,974 /24 blocks in 65 university groups, where the addresses in each group are likely in one location. Our method correctly identified the majority of the addresses in each group as co-located.

We also used our delay-based method to assess the /24 block co-locality assumption. We examined a large dataset of 1.41M /24 blocks and showed that more than 17% appear to be multi-location blocks. This outcome disagrees with the common assumption of /24 block co-locality in many geolocation services, suggesting we need to identify which addresses are in

one location instead of making such assumptions. We found that the majority of the blocks identified as multi-location belong to large ISPs in countries with rich Internet connectivity such as the United States and other countries in Western Europe.

Our work can be part of a system to evaluate geolocation databases such as MaxMind. Such a system would likely deploy a long-lived active measurement infrastructure and compare results with both free and commercial geolocation databases. A long-lived system can also track the movement of IP address blocks as they get traded, which we study in Chapter 5. We believe that our method can be used to conduct longitudinal studies to identify consistently co-located groups of IP addresses over time. A geolocation system can use this information to geolocate a set of co-located IP addresses as a unit by a few of its representatives, effectively trimming the number of geolocation targets without losing accuracy.

Chapter 5

Delay-based Identification of Internet Block Movement

This thesis focuses on characterizing the IP addresses co-locality and movement using latency measurement to enable efficient IP geolocation that maintains up-to-date IP-to-location mappings. Chapter 4 presented a method to assess the co-locality of IP addresses, allowing us to identify co-located addresses that we can geolocate as a unit and avoid geolocation error that could result from incorrect co-locality assumptions. In this chapter, we turn our focus to identifying movement of IP address blocks (or address blocks).

Some IP blocks occasionally change their physical location, such as when blocks are transferred to different organizations, or repurposed within an organization. We expect that the majority of the Internet blocks do not move since most organizations reside in a well-defined physical space and tend to keep their assigned addresses within that space. Therefore, full re-geolocation is inefficient to maintain up-to-date geolocation data when only a fraction of the Internet blocks requires location updates from time to time.

To support our thesis statement (§1.2), we show that we can leverage latency measurement to determine the location-stability of address blocks to detect when a geolocation update is required. Identifying address blocks that move allows an IP geolocation system to trigger re-geolocation as needed to maintain up-to-date location-mappings and avoid unnecessary, expensive full re-geolocation of the whole Internet (§2.5).

To detect block movement, we track all ping-responsive IPv4 /24 blocks from a handful of globally distributed vantage points (§5.2.1). Our method looks for significant and persistent changes in the latency state of a /24 block observed from multiple vantage points around the same time as an indication of block movement (§5.3.3). Using the proposed method, we show that only a small fraction of the responsive Internet blocks occasionally moves (§5.5.1). We

estimate around 2.1% of the 3.77M /24 blocks we studied have changed location at least once in the last 3 months of 2018 (2018q4). We validate a random sample of blocks we identify as moving and confirm 80% (41 of 51) through traceroutes (§5.5.2).

5.1 Introduction

Chapter 2 presented several IP geolocation approaches. Regardless of the geolocation approach used, the result only provides a snapshot of the current IP-to-location mappings. Some IP blocks occasionally move to a different location, for example, when transferred to a different organization, or reassigned within an organization. Previously estimated location of a block that has moved becomes outdated and needs to be updated. It is possible that IP block movement is now more frequent due to the exhaustion of the registries' IPv4 address pools that serve the global demand [88, 104].

The geolocation accuracy can have a significant impact on Internet applications that utilize geolocation information such as Video on Demand (VoD) services that use geo-blocking to limit or block access to their content based on users' location [40, 83]. IP block movement can degrade geolocation accuracy if not detected and timely addressed. This chapter looks to identify Internet block movement to help a geolocation system maintain an up-to-date IP geolocation without having to perform frequent geolocation for the whole Internet.

A geolocation system needs to identify when a block moves and then update its location. One may obtain information about block movement from the Regional Internet Registries (RIRs) transfer reports, which may report IP address range transfers between organizations [7,9]. However, the reported date of transfer does not necessarily reflect when the block actually appears in a new location (§5.5.3). More importantly, these reports do not include information about ISPs internal reassignment of blocks to other locations. *Measurement-based* geolocation methods (§2.3) can maintain up-to-date geolocation if applied continuously, but these methods can be intrusive and inefficient when applied continuously to the entire IP address space.

The primary goal of this chapter is to define a method to identify when blocks (/24 IPv4 prefixes) move in order to help a geolocation system to maintain up-to-date IP-to-location mappings without requiring full re-geolocation. The identification of a block movement tells a geolocation system it is time to re-run geolocation to update the block location. To achieve this goal, we propose a *delay-based* method that monitors ping measurement to visible /24 blocks from 6 globally distributed vantage points. We show that these measurements can identify block movement and are inexpensive enough to run continuously. Our method identifies movement by observing persistent changes in the latency state of a /24 block from multiple sites around the same time.

The first contribution of this chapter is defining an efficient method to identify IP block movement from delay measurements observed via a small number of vantage points. Our second contribution is the application of our method to a dataset of 3.77M /24 blocks, showing that 2.1% of them experienced movement during the last 3 months of 2018.

5.2 Datasets

This work uses a USC ICMP echo-request (ping) data coverage to look for block movement in about 4M /24 blocks of the responsive Internet (§5.2.1). We then use two datasets from CAIDA to validate a sample of our block movement findings: the IPv4 Routed /24 Topology Dataset (§5.2.2), and the Internet Topology Data Kit (§5.2.3).

5.2.1 Latency Information from the USC Internet Outage Data

Our method evaluates delay measurement to /24 IP blocks over time, so we require Internet-wide data that contains latency estimates. We extract latency estimates from publicly available measurements taken for Internet outage detection [97] using Trinocular [101]. This data is available to researchers at no cost; we obtained it from USC.

Trinocular scans IP addresses in about 4 million responsive IPv4 /24 network blocks using ICMP echo-request messages. (The target list of blocks is updated periodically using long-term

history data from Internet censuses [41].) Each /24 block is probed every 11 minutes, one or more probes taking place, often stopping after the first successful probe returns an echo-reply. Each block is therefore probed about 130 times a day. Successful replies include the round-trip time; we ignore unsuccessful probes. Scans rotate through different addresses in each block over time.

Trinocular collects data from six vantage points (VPs) positioned around the world. We use data from all of the six vantage points collected during October through December of 2018 [99]. The VPs identifiers and locations are: **c** (center of the U.S in Ft. Collins, Colorado), **e** (east coast of the U.S. in Arlington, Virginia), **g** (Athens, Greece), **j** (Tokyo, Japan), **n** (Utrecht, Netherlands), and **w** (west coast of the U.S. in Marina del Rey, California).

We estimate block latency each day from its daily observations (§5.3.1). Each attempt tries up to 15 addresses and reports latency only if one replies. We determine a block's latency state status only for days when 3 or more VPs each have 10 or more latency observations, which we refer to as *determination-valid* days. About 156k (3.9% of all blocks in the ping dataset) do not have any determination-valid days and around 41k (1%) have 9 or less such days. In the remainder of this chapter, we use the remaining 3.77M blocks (95.1% of all blocks) with 10 or more determination-valid days.

5.2.2 Paths from the CAIDA UCSD IPv4 Routed /24 Topology Dataset

To examine the relationship between observed changes in latency estimates and routing changes, we use the *CAIDA UCSD IPv4 Routed /24 Topology Dataset* (henceforth referred to as *CAIDA-topology* dataset in this chapter) [14] We use historical traceroute data from the same period as our ping dataset (§5.2.1).

The CAIDA-topology traceroute measurements are collected using around 152 *Ark monitors*, globally distributed in 52 countries. These monitors work as a team to probe randomly selected IP addresses in every routed /24 prefix. Only a single random destination in a /24 prefix is probed every 48 hours by only one of the monitors.

To get observations from locations near our six vantage points, we first identified active Ark monitors that are within 50 km of our probes. We found 15 such Ark monitors close to 4 of our vantage points. We used the closest Ark monitors available for the other two vantage points; the Ark monitor *wbu-us* at Boulder, CO for the VP_c at Fort Collins, CO, U.S., and *sof-bg* at Sofia, Bulgaria for the VP_g at Athens, Greece.

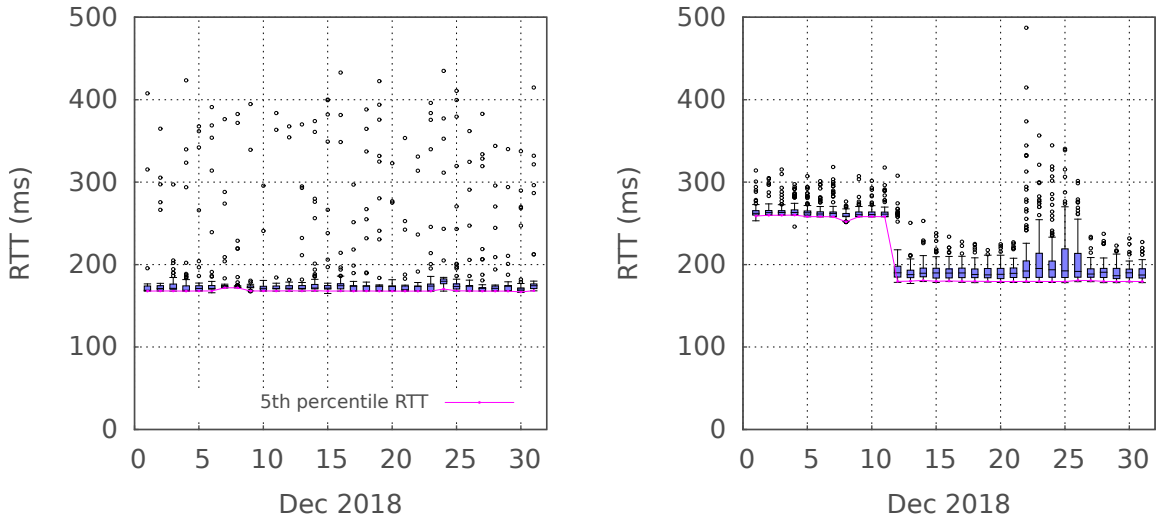
Ideally, we would like to compare one monitor's traceroute measurements soon before and after a block sees a change in latency. We first identify blocks with changes in their latency according to §5.3. For a given change, we select one monitor's measurements if they satisfy the following criteria: First, the measurements should be from the *consistent-RTT* periods (i.e., periods with no other changes detected in them) before and after the change. Second, only monitors with at least one measurement before the delay change, and another after it are used. If a monitor has multiple measurements that satisfy these two criteria, we select the closest in time to the observed change.

We do not always find relevant traceroute measurements that satisfy our criteria. Each routed IPv4 /24 prefix in the CAIDA-topology dataset is probed only once every 2 days, by only one of the Ark monitors, which may or may not be one of the 17 Ark monitors we selected. Applying the selection methodology to blocks we identify with delay changes shows that around 62% of the changes have relevant traceroute measurement.

5.2.3 Paths from the CAIDA Internet Topology Data Kit

For a more comprehensive routing path comparisons, we augment the measurements of the CAIDA-topology dataset (§5.2.2) with data from *CAIDA's Macroscopic Internet Topology Data Kit* (ITDK) [13]. We use the ITDK data to map any traceroute hop IP address to its router-level node, AS, and location. In this chapter, we use the ITDK 2018-03 dataset, the closest public ITDK dataset in 2018 to our ping dataset.

The ITDK data has two router-level topologies, derived with different alias resolution tools. We used the one derived with MIDAR [72] and iffinder [71] tools, which CAIDA reports as the



(a) VP *w* and block 129.16.71.0/24.

(b) VP *n* and block 45.226.48.0/24.

Figure 5.1: Box plots and 5%ile RTT for sample /24 blocks. Dataset: 2018q4.

more accurate, although with less coverage. The dataset extracts the IP addresses of intermediate hops that appear in traceroute measurement performed using the Ark infrastructure.

AS assignments are derived using RIPE and RouteViews BGP tables and Regional Internet Registries (RIR) delegations. The geolocation is derived at the router-level using different sources that include hostname mapping, information from known Internet eXchange (IX) point, and MaxMind's free GeoLite City database. A router is assigned a location only when all of its identified interfaces are individually geolocated to the same location.

5.3 Methodology: From Block Latency to Block Movement

Our methodology begins by processing observations of block RTT to get a stable estimate of its latency (§5.3.1). We then show examples of the patterns in these estimates that indicate block movement, congestion, and routing changes (§5.3.2). Our detection method searches for these patterns to detect block movement (§5.3.3).

5.3.1 Stable Estimation of VP-to-Block Latency

To get a stable estimate of block latency we must filter through measurement noise due to network congestion, route changes, and other transient network effects. We use *5%ile of all daily RTT observations* (or just *5%ile-RTT*)—a measure at which 5% of the observations lie below— as our stable estimator of the block's latency. This near-minimum estimate of RTT filters out queuing delay from congestion while avoiding outliers.

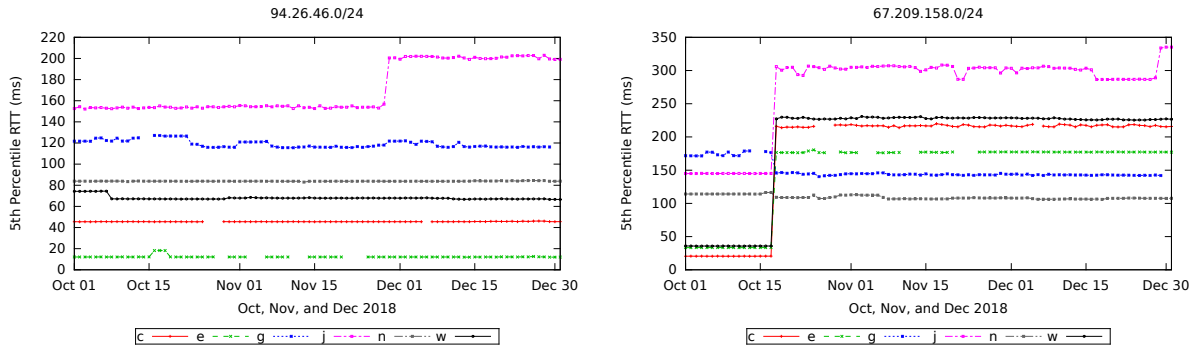
Figure 5.1 depicts the box plots of two sample /24 blocks daily RTT observations and their 5%ile (the magenta line). Each plot depicts daily observations for one VP over one month. Each box shows the interquartile range (IQR), with RTT observations with the lower and upper quartiles (25%ile and 75%ile) with an interior line showing the median. The lower (and upper) whiskers show the lowest RTT still within (1.5× IQR) of the lower (or upper) quartile, and circles show outliers beyond the whiskers.

First, we observe that quartiles are quite tight, suggesting that we can filter outliers. Moreover, most outliers are above the upper whiskers, suggesting that the 5%ile will be close from minimum RTT and therefore speed-of-the-Internet latency. Having established this estimator, we report only the 5%ile-RTT in later sections and omit quartiles and outliers.

Second, we can see that these two VPs show different latency fingerprints. For Figure 5.1a, while RTT observations vary some throughout the day, the daily 5%ile is relatively stable over time. In §5.5, we show that this *5%ile-RTT stability* is common for most blocks. By contrast, the block in Figure 5.1b also shows 5%ile-RTT stability, but with *two modes*, values before and after December 12 each show a different common value, and latency drops by about 78 ms on the 12th. This change indicates either routing or location change, and in §5.3 we describe our algorithm to identify blocks movement by detecting such changes across *multiple* VPs.

5.3.2 Common Patterns in IP-Block Latency

For each day, the 5%ile-RTTs from 6 VPs defines the that block's *latency state* (or just *block latency*). We look for patterns of change in block latency that indicate block movement. Here we



(a) Latency change in one VP suggests a routing change.

(b) Latency change in all VPs suggests block movement.

Figure 5.2: Example latency states for blocks showing routing change (left) and movement (right). Dataset: 2018q4.

show sample patterns drawn from blocks in 2018q4. We use the insights from these examples to define our algorithm in the next section.

Figure 5.2 depicts the 5%ile-RTTs for two sample /24 blocks, where each line represents the data from one vantage point. For the block in Figure 5.2a, the daily observations are mostly consistent up to November 29 for all 6 VPs. This RTT-stability suggests the block's location is fixed during that period. This pattern is common in most blocks in our ping dataset, as we show in §5.5. We expect that a change in a block location would affect the observations of multiple VPs. Only VP_j observed a significant change on November 29, suggesting an event that affected only that VP and not the block (e.g., a routing change on the path from VP_j to that block). We also see other, less significant changes in delay, including VP_n on October 8, VP_e on October 16, and VP_g on several occasions. Many small changes at different times suggest routing changes or persistent congestion, not block movement. Our algorithm in §5.3 looks for persistent, significant latency change to indicate movement and filter out other effects.

Figure 5.2b shows an example of a /24 block with a significant change in the block latency observed by multiple vantage points on October 18. Some of the vantage points experienced an RTT increase (VP_c, VP_e, VP_j, and VP_w), while VP_g observed a decrease, and VP_n observed an insignificant change. We also note that the new block latency after October 18 is persistent

through the end of December. This pattern indicates that the block likely moved to a different location on November 18, 2018.

5.3.3 Identifying Block Movement from Latency Measurements

Building on 5%ile-RTT estimates from 6 VPs, and common patterns to look for, we now present *our algorithm for block movement*.

Our algorithm has four steps, each confirming the block is suitable for continued analysis. First, we determine blocks with sufficient latency observations as *determination-valid*, then we look for *changes in VP latency*, that show *VP agreement*. Our final check is *persistence* of the change. We review each of these steps below.

Our first step is to confirm that the block has *determination-valid* days. A block's one day worth of measurement is determination-valid when, over the course of that day, it has enough VPs, each with enough observations that we can draw statistically strong conclusions. We require that each VP have at least 10 latency estimates, so that that VP's 5%ile is valid (not mislead by transient network conditions). We require observations from at least 3 VPs, so that we can confirm movement and not just a route change affecting one path. Since each VP makes about 130 attempts to measure latency and we have 6 VPs, these requirements (of 10 estimates per VP and 3 VPs) allow for substantial downtime or measurement error.

Our next step is to look for *changes in VP-to-block latency*. In §5.3.2, we showed stationary blocks see some variation in daily 5%ile-RTTs. We consider latency from a VP to the block to have a *significant change* if the change exceeds some threshold T compared to the long-term average (one week). We use a threshold of a 9% change in latency, as determined from ROC analysis from training data (§5.4.3).

We use *VP agreement* to filter out routing changes. Physical movement usually affects all VPs, but changes in Internet paths will change latency between a VP and the target block, and are unlikely to affect all VPs. We consider an *agreed-latency-change* to occur when there are VP-block latency changes by at least half of the VPs (3 or more). We do not demand the agreement

of all VPs as some might not have VP-latency-valid days around the time a change happens. Moreover, some VPs may see insignificant changes concerning the delay-change threshold criteria.

Finally, we require that the latency change is *persistent*. IP blocks are unlikely to move frequently or for short periods. Therefore, latency changes caused by a physical block movement will likely persist more than just a few days. We compute the duration of an agreed-latency-change as the number of days until we observe another change, or until the VPs agreement heuristic is broken (i.e., we no longer have 3 or more VPs with significantly different block latency to that before the agreed-latency-change). In this work, we focus on agreed-latency-changes that persist for at least one week as a strong indication of block movement.

When a block meets these four criteria we consider it a *movement candidate*.

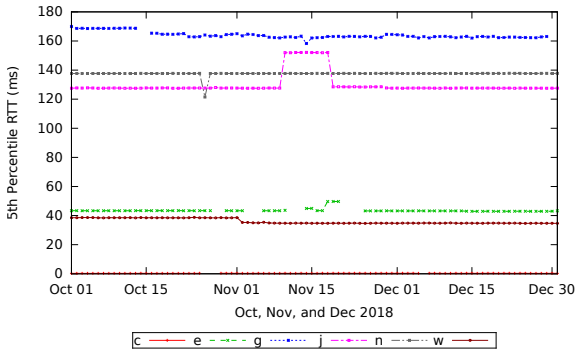
5.4 Controlled Experiments with Synthetic Data

To evaluate our method with known ground truth, we next describe how we simulate block movement (§5.4.1), build a test dataset from synthetic movement of real observations (§5.4.2), and use this to select optimal parameters for our method (§5.4.3).

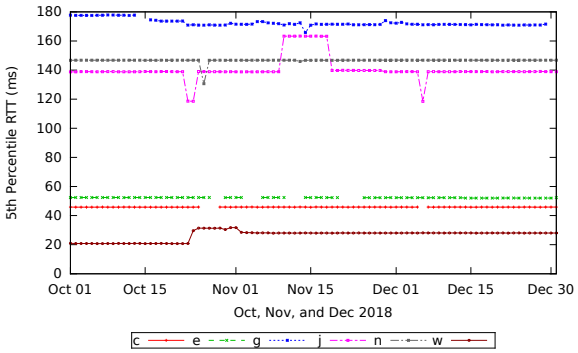
5.4.1 Simulation of Block Movement

We simulate block movement by replacing a block's RTT data in a selected range of days with data from another block at a different location. (We use only latency, the addresses involved are unimportant.) We select start and end randomly, with end at least 7 days after start.

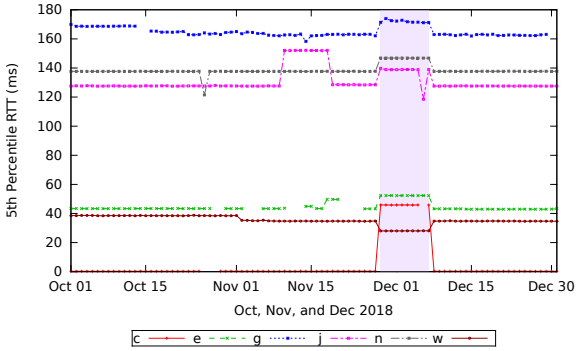
Figure 5.3a and Figure 5.3b show 5%ile-RTTs of blocks about 700 km apart (in Fort Collins, Colorado and Logan, Utah). They show some variation in latency over the observation month, but neither is identified as moving by our algorithm. In Figure 5.3c, we create a *synthetic* block that moves from Colorado (the base data) to Utah from November 28 to December 7 (the shaded region) by replacing that period of data.



(a) Block 129.82.44.0/24, Fort Collins, CO.



(b) Block 129.123.54.0/24, Logan, Utah.



(c) A synthetic /24 block with movement in the shaded period.

Figure 5.3: Real blocks (dataset: 2018q4) combined to make a synthetic block with known movement (dataset: synthetic).

5.4.2 Building a Dataset with Synthetic Movement

Next, we build a dataset with synthetic movement of block. We begin with 60 /24 IP blocks, each with websites from a different university. We verify that each block appears to be physically at its university (and not outsourced to a third party) using *WHOIS* information, and reverse DNS names, checking that the name indicates the university. We select university blocks because academic institutions have known locations and often self-host, suggesting their blocks are at static, known locations.

We select geographically distributed blocks: 8 universities in Africa, 9 in Latin America, 10 in Asia, 13 in Europe, and 20 in North America. We use the *GeoNames geographical database* [46], to identify the geographic location of each block from its university.

Table 5.1: Accuracy from method to the university blocks. Dataset: 2018q4.

Delay-change threshold	**3%**	**5%**	**7%**	**9%**	**11%**	**13%**	**15%**
False Positives	13	4	3	2	1	0	0
True Negatives	47	56	57	58	59	60	60

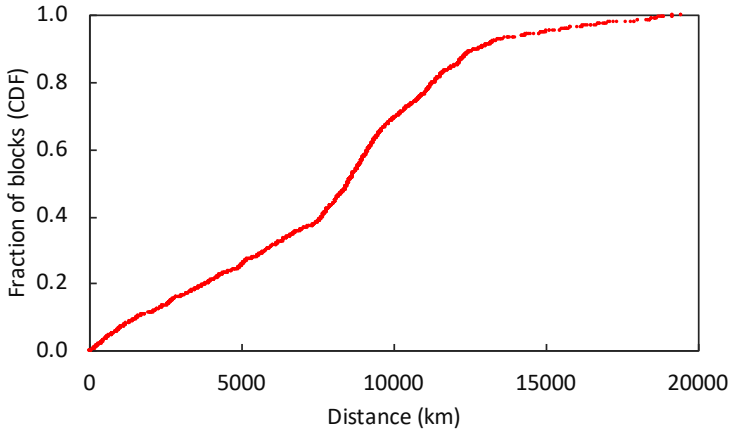


Figure 5.4: Distribution of distances in all 1,540 block pairs with known movement. Dataset: synthetic.

Since our goal is to identify blocks that move, we consider all 60 university blocks as *negative instances* of this category. We verify the blocks do not cause false positives in our algorithm by applying movement identification with *delay-change* thresholds from 3% and 15% (with 2% increments). Table 5.1 shows false positives (FP) and true negatives (TN) per delay-change threshold. We drop a delay-change threshold of 3% as too sensitive, since with it, 13 blocks (21.7%) are misclassified as moving. Larger thresholds show a few or no false positives. We identify 4 blocks that are false positives at higher thresholds, so we eliminate them from use in synthesis.

Using the remaining, verified 56 blocks, we create all 1,540 possible pairs where we insert data from one block into another to simulate movement (as in Figure 5.3). Figure 5.4 shows the distribution of distances of the 1,540 block pairs in the synthetic dataset. Most of the pairs (85%) show uniform distances from 0 to 12,000 km.

Table 5.2: Accuracy for different delay-change thresholds from known movement. Dataset: synthetic.

Threshold	TPs	FNs vs distance (km)					
		any distance (1540)	0-100 (7)	101-200 (11)	201-300 (12)	301-400 (8)	401-500 (19)
5%	1516	24	5	5	5	1	2
7%	1505	35	5	6	5	1	3
9%	1495	45	5	6	7	2	4
11%	1474	66	5	7	8	4	8
13%	1449	91	5	8	11	5	11
15%	1434	106	5	8	11	5	14

5.4.3 ROC Analysis

We now use this synthetic dataset of 1,540 blocks with known movement (§5.4.2) to select optimal parameters for our algorithms with ROC-curve (Receiver Operating Characteristic) analysis.

Table 5.2 shows how many blocks move (TPs) or do not have detected movement (the FNs at any distance) *vs.* different delay-change thresholds. The table also compares FNs against distance ranges below 500 km, since short distance movement is more challenging. The number in parentheses under each range is the count of instances we have in that range. Geographically closer blocks show smaller differences in latency, and our method performs better as distances increase. Smaller thresholds seem to have fewer false negatives since they are more sensitive.

We use the ROC-curves to select the delay-change threshold that balances true positive rate (TPR) with false positive rate (FPR). We compute the TPR and FPR at various thresholds from the confusion matrix of the analysis of university blocks (Table 5.1) and the synthetic-moving blocks Table 5.2. Figure 5.5 shows the ROC curve for the TPR against FPR at different thresholds (shown next to the marks on the graph). We see that a threshold of 9% yields good TPR against FPR results (97% and 3.3%), allowing for detecting most of the moving blocks while reducing false positives over the evaluation datasets. We use this threshold in the remainder of this chapter.

We next test the sensitivity of our method and the selected delay-change threshold (9%) on a more challenging subset of the synthetic blocks. Rather than using all possible 1,540 synthetic combinations, we perform the ROC-curves on synthetic pairs of distance less than 4,500 km

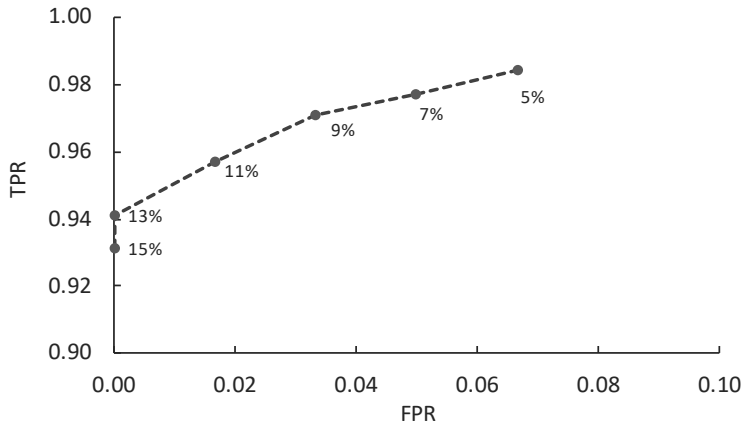


Figure 5.5: ROC curve showing true and false positive rates over blocks with known movement. Annotations next to points show the threshold, and the y-axis does not start at zero. Dataset: synthetic.

(comparable to the horizontal width of the African continent or a large country such as the U.S.). We find 23% of the synthetic blocks (360) that satisfy this criteria.

Table 5.3 shows the true positives and false negatives of applying our method to the 360 synthetic blocks with known movement within 4,500 km. The results are consistent with those for all 1,540 synthetic blocks (Table 5.2), showing that smaller thresholds are better at detecting blocks with smaller movement.

Table 5.3: Accuracy at different delay-change thresholds from known movement at *continent scale*. Dataset: synthetic.

Threshold	**TPs**	**FNs**
5%	342	18
7%	335	25
9%	328	32
11%	305	55
13%	291	69
15%	281	79

For the ROC-curves analysis, we compute the TPR from the confusion matrix of synthetic-moving blocks analysis (Table 5.3), and the FPR from the previous analysis of university blocks (Table 5.1). Figure 5.6 shows the ROC curve for the TPR against FPR at different thresholds (shown next to the marks on the graph). Although the movement scale of the 360 synthetic

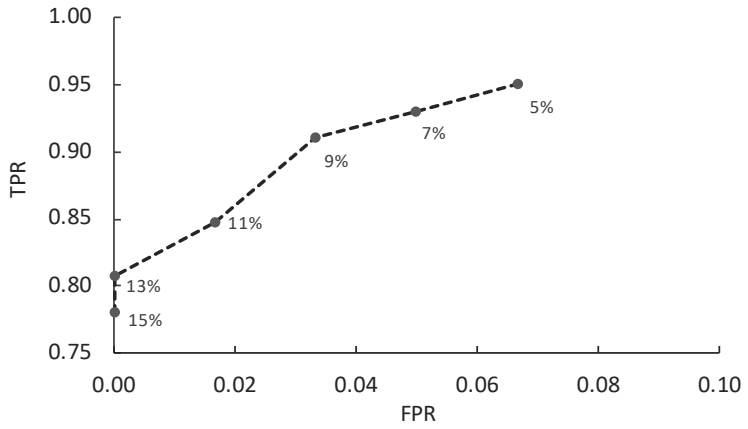


Figure 5.6: ROC curve showing true and false positive rates over blocks with known movement within a continent scale. Dataset: synthetic.

blocks is far smaller than that of all the 1,540 blocks, our method still achieves good results that are consistent with those in Figure 5.5. We see that a threshold of 9% again yields good TPR against FPR results (91% and 3.3%), confirming our results over all the synthetic blocks.

5.5 Evaluation with Real-World Data

Having defined our methodology with controlled experiments, we now apply our method to real-world data for about 3.77M /24 blocks to understand the Internet (§5.5.1) and verify real-world block movement (§5.5.2 and §5.5.3).

5.5.1 Applying our Method in the Wild

We next apply our method to Internet-wide data (§5.2.1) to identify blocks that move. Our goal is to identify how many blocks move during 2018q4. We do not expect many blocks to move, since most organizations have a fixed physical location, and assignment of addresses is often stable.

We summarize our results in Table 5.4, using our method with a delay-change threshold of 9%. We see that 78.7k (2.1% of the 3.77M /24 we consider) move at least once during the last 3 months in 2018. These results show that our algorithm can identify a subset of moving blocks that IP geolocation services should review, saving 98% of the effort of checking everything.

Table 5.4: Block movement from Internet-wide data. Dataset: 2018q4: threshold: 9%.

ping-dataset blocks	~4M	
determination-valid Blocks	3.77M	100%
consistent *5%ile-RTT*	3.33M	88.3%
agreed-latency-change	441k	
short	362k	9.6%
indicate movement	78.7k	2.1%

Most of the blocks in our ping dataset showed consistent 5%ile-RTT. About 3.33M (88.3%) show consistent latencies (no agreed-latency-change) over the in 3 months, meaning at no time there was a significant latency change agreed upon by 3 or more VPs. Another 362k blocks (9.6%) see one or more short agreed-latency-changes, but return quickly. These short changes suggest transient network events such as routing change or congestion, not movement.

5.5.2 Validating Block Movement with Historic Traceroutes

Next, we use the CAIDA-topology (§5.2.2) and ITDK (§5.2.3) datasets to confirm our block movement findings with router-level path information.

We start by finding relevant traceroute data for a moving block, as described in §5.2.2. We then use the ITDK dataset to map traceroute hops (interfaces) to routers (for de-aliasing), ASes, and locations (§5.2.3). We require the AS of the last identified hop to match the AS of the target block (the *AS-criteria* for traceroutes).

We consider two traceroute hops to match if they map to the same router. Comparing traceroutes from before and after a latency change can show: (a) near-identical routing paths, indicating no movement; (b) change in intermediate routers only, suggesting the latency change is due to a routing change; (c) change towards the target block, suggesting a block has moved. We confirm (c) when the penultimate hop changes AS.

We evaluate 100 randomly chosen blocks from the 79k that we identify as moving, and show the results in Table 5.5. Of those 100 blocks, 60 have traceroutes, and 51 traceroute to the target AS and so can be used to test work. We find 10 of these 51 have near-identical traceroutes or show intermediate routing changes—these blocks are likely false positives due to congestion or

Table 5.5: Validation of block movement with traceroute data. Datasets: 2018q4, ITDK, and CAIDA-topology.

candidates	79k	
random samples	100	
no traceroutes	40	
with traceroutes	60	
misses AS-criteria	9	
passes AS-criteria	51	100%
no or int. change	10	20%
near end change	41	80%
country change	26	
city change	8	
no city data	7	

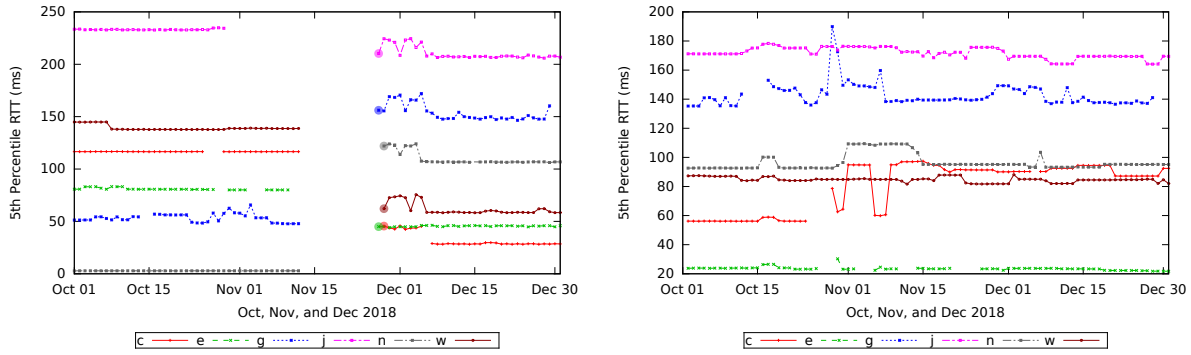
Table 5.6: Validation of block movement with transferred blocks. Datasets: 2018q4 and RIR reports.

ARIN reported transfers	2,416	
no latency	2,400	
have latency	16	
lack before or after	13	
lack before	5	
lack after	8	
have before-and-after	3	100%
confirmed move	2	67%
did not move	1	33%

routing changes. The remaining 41 blocks show traceroutes with different penultimate ASes, suggesting movement—about 80% show true positives. Geolocation confirms 26 (of 41) map to changed country and 8 changed city, confirming movement. The remaining 7 changed AS but we could not confirm movement because they lacked city-level geolocation.

5.5.3 Validating Block Movement with Transferred Blocks

We next examine blocks between Internet regions (defined by RIRs, the Regional Internet Registries: ARIN, RIPE, APNIC, LACNIC, AfriNIC). We examine the 74 IP ranges (2,416 /24s) that ARIN reported as inter-RIR transfers in 2018q4 [9]. We have latency data for only 16 /24s in that set, probably because transfers are often of previously unused blocks [10, 104].



(a) Block 185.169.108/24.

(b) Block 69.94.100/24.

Figure 5.7: Latency data for two inter-RIR transferred blocks. Dataset: 2018q4.

Out of the 16 blocks, 13 cease responding to ICMP midway through the quarter, suggesting they were transferred but have not yet resumed service. Although we expect these blocks are in the process of moving, our algorithm cannot detect movement until they resume service, so we do not consider them further. There are many such blocks (130k in the quarter); such blocks deserve monitoring for when they resume service and we should reevaluate geolocation.

Examining the remaining 3 blocks: our method identifies 2 as moving. Figure 5.7a shows the 5%ile-RTTs for one of them, block 185.169.108/24. (The other block is 185.169.109/24 and shows similar results. Both are from BGP prefix 185.169.108/22 and are announced by AS395855.) The 185.169.108/24 block was transferred from an organization in the Netherlands (under RIPE) to a recipient in the U.S. (under ARIN). This block responds to ICMP through November 12, goes silent, and then resumes service November 27. According to ARIN's report, the transfer was effective on October 25, 2018. After the block resumes, observed latency does not stabilize until Dec. 5. We hypothesize that the new block operators were debugging routing over the first week of December. The 5%RTTs before and after the gap are quite different. The data is consistent with this block moving to a new location, as found by our algorithm. It is surprising, though, that it was responsive after the transfer date; perhaps the paperwork preceded routing changes.

The third block is 69.94.100/24, with 5%-ile RTTs shown in Figure 5.7b. This block was transferred between two different cloud hosting services, but one under ARIN and the other APNIC. Our algorithm do not show it moved, we see some fluctuation in latency around the reported

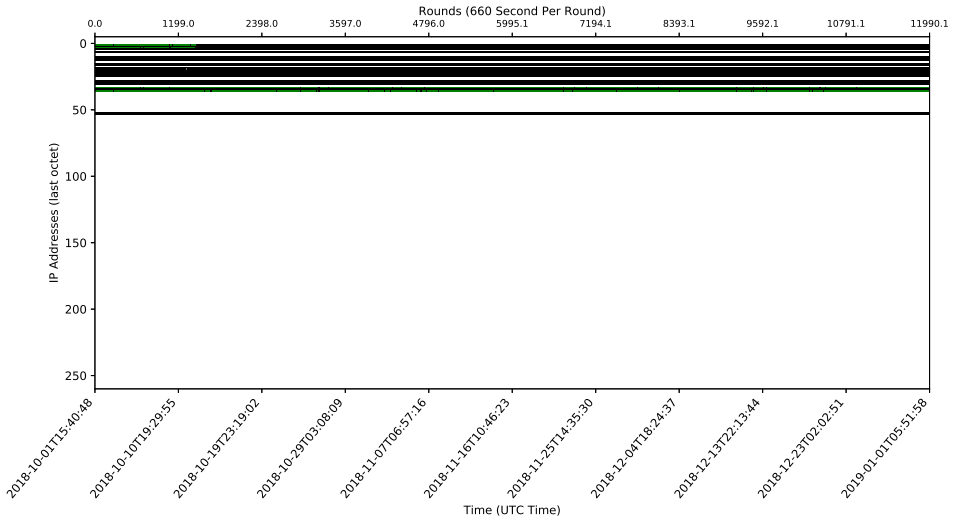


Figure 5.8: ICMP responses for block 69.94.100/24, showing similar behavior over all three months.

transfer date (November 1), but not persistent changes for most of VPs, and no interruption in service. We hypothesize that this block was transferred with hardware in a data center, so although the RIR responsible for the address space changed, we believe the block did not move.

To confirm that the block responded identically before and after the transfer date, Figure 5.8 shows the raw ICMP responses we observed. In the figure, each green dot is a positive response, black are non-responses, and white are addresses that are not probed.

We see that two addresses (last octets 33 and 35) replied consistently through the entire three months, including times both before and after the RIR transfer. Two other addresses (last octets 1 and 3) stopped on 2018-10-12. Continuous addresses are consistent with the block changing administrative responsibility, but *not* actually moving.

This section complements our prior validation with traceroutes (§5.5.2) with validation with documented change of allocation. Although we have before-and-after data for only 3 blocks, our data demonstrates movement in two cases and suggests non-movement in the other.

5.5.4 Movement over Time

To confirm the fraction of blocks we identify as moving in 2018q4 is representative and not an outlier, and to get a measure of the rate of movement over time, we apply our algorithm

Table 5.7: Block movement from Internet-wide data. Dataset: 2019q1: threshold: 9%.

Ping dataset blocks	~4M	
determination-valid Blocks	3.82M	100%
Consistent *5%ile-RTT*	3.49M	91.3%
agreed-latency-change	333k	
Short	268k	7.2%
Indicate movement	65k	1.7%

to data from a different quarter, the first quarter of 2019 (2019q1) [100]. We then compare the results with those of 2018q4 (§5.5.1).

Table 5.7 shows the results for 2019q1. The total number of /24 blocks probed is just a few hundreds less than 4M. Around 3.82M of these blocks are determination-valid blocks (§5.2.1), 3.59M of which are also determination-valid blocks in the 2018q4 dataset.

We identify about 65k (1.7% of the determination-valid blocks) to have moved during 2019q1, around 14k blocks fewer compared to 2018q4 (Table 5.4). We find 11k blocks as common movers from the two quarters. Overall, the results over the two quarters are consistent. We again observe that most of the blocks are RTT-stable and identify a small fraction of the responsive blocks as moving.

As future work, we plan to extend our study of block movement using data from additional quarters. This longitudinal study could help quantify the rate at which blocks move over time, showing which blocks are location-stable and which are more dynamic.

5.6 Conclusions

We have shown an efficient method that identifies the *movement of IP blocks* using existing ICMP scans, based on changes in the latency "fingerprint" from multiple, distributed observers (§5.3.3). We validate our approach by confirming movement through traceroutes and information about Internet registration re-allocations.

This chapter provides additional evidence to support our thesis statement (§1.2) that we can use latency observations to characterize the location-stability of the IP address blocks. We showed that most of the responsive Internet blocks are RTT-stable, suggesting location-

stability (§5.5.1). Therefore, a geolocation service does not need a full re-geolocation to keep up-to-date geolocation. We identified about 2.1% of Internet blocks as moving over a quarter (2018q4), and 1.7% for another quarter (2019q1), suggesting our approach will help IP geolocation providers to keep their data up-to-date efficiently. As future work, we plan to use our method to study the location stability of IP blocks over extended periods to dictate the required frequency of their measurement and location updates.

Chapter 6

Delay-based Identification of Internet Cellular Blocks

This thesis proposes methods that leverage latency observations to assess IP addresses co-locality (Chapter 4) and detect address block movement (Chapter 5) to enable efficient application of the state-of-the-art IP geolocation methods. This chapter demonstrates how address blocks in cellular networks show distinct latency patterns and present a compelling case to study as they may present a challenge for delay-based geolocation and other delay-based methods, such as those proposed in our thesis. Our primary goal in this chapter is to define a client-independent method to identify cellular blocks, a step toward studying their geographic properties and effects on delay-based methods in future work.

Mobile handsets today are the dominant devices to access the Internet. Previous work has shown that addresses in cellular networks can be challenging for geolocation (§2.6). For the delay-based methods we propose in this thesis, identifying blocks used for cellular access allows a better interpretation of their latency observations and provides an opportunity to study their co-locality and movement over time, studies we look to do as future work. Identifying cellular blocks also allows studying their traffic trends and impact on the Internet and diagnosing performance issues.

Previous studies of cellular networks rely on network operators or user collaboration to identify cellular blocks (§2.6). In this chapter, we propose a delay-based method to identify cellular blocks without requiring such collaboration (§6.3.3). We identify network type by the *variation* in RTT in each block of addresses—cellular networks show very large variation, whereas wired and WiFi show little to modest variation (§6.3). We measure a block's daily RTT variability using the *Interquartile range* (*IQR*) of the RTT samples and use it to devise a method to distinguish between cellular blocks and the other block types.

6.1 Introduction

The growth in the number of devices connecting to the Internet via cellular wireless technologies and their corresponding traffic is astounding. A mobility report from Ericsson estimates 7.8 billion mobile subscriptions worldwide in 2017 and projects an increase to 8.9 billion in 2023 [37]. The report also predicts 3.5 billion cellular IoT connections in 2023, a growth that would impact many industries and businesses. Since fall 2016, the majority of web accesses are reportedly from non-desktop devices [120]. Cisco reports that wired devices contributed only half of the total global IP traffic in 2016, and forecasts the wired fraction to decline to 37% by 2021 as wireless and mobile increases [26].

While these reports estimate wireless growth, or measure web use, we would like to *directly measure cellular wireless use on the Internet.* Identifying addresses in the public Internet that connect via cellular networks can help quantify this trend and its impact on the Internet. Network operators and users may collaborate to identify wireless IP prefixes. However, it is essential to identify these prefixes independently when such collaboration is not available.

Identification of cellular addresses can benefit Internet service providers, policy makers, and network researchers. This information can help content providers identify performance issues and improve their service quality. It also gives policy makers valuable context to understand use of the networks that they regulate. As one example, understanding the relative reliability of wired and wireless networks during disasters can help assess Internet resilience [59]. For researchers who want to characterize the mobile Internet, identifying cellular addresses is a first step on this goal. Delay-based applications such as measurement-based IP geolocation [49, 70, 128] can also benefit from cellular identification as knowledge about wireless can lead to better interpretation of delay observations and thereby improve geolocation.

Independent identification of wireless IP addresses is challenging. Previous studies of wireless network performance and traffic assumed prior knowledge of which hosts access the Internet through wireless networks rather than identifying them [87, 108, 111, 117, 131, 133]. Some of these studies use applications installed on end-user devices, while others use information

from carriers or operators, approaches that limit the scope of such studies. Others examine web access by browser type, but only provide aggregate statistics [120]. Prior work has shown that cellular networks identification needs to be at a level smaller than an autonomous system [109]. Our primary goal in this chapter is to define a method to identify cellular IP blocks without external information from operators or users, that is, using only measurements taken on the public Internet.

We propose a new method to identify cellular blocks based on active measurements of public IP address blocks. We show that we can distinguish between cellular blocks and the other block types by their Round Trip Time (RTT) patterns when they are probed repeatedly over time from a vantage point (VP). We show that cellular blocks exhibit RTT patterns different to those of fixed-line and WiFi blocks . Our approach requires the IP blocks to be responsive to ping probes but otherwise requires no cooperation from users or operators.

The first contribution in this chapter is to define a new, delay-based method to classify blocks as *cellular*, *strictly WiFi*, or *mixed* (wired or WiFi). We find robust classification thresholds to distinguish between the three classes merely based on blocks' daily RTT patterns. Our method is most successful at identifying cellular blocks since their RTT patterns are the most distinguished. Our second contribution is to apply this method to most of the public Internet, reporting on about 3.72M responsive IPv4 /24 blocks data from September 2017. We identify about 169k blocks (4.6%) as cellular, 23k (0.6%) as strictly WiFi, and a majority of 3.5M (94.8%) as mixed.

6.2 Datasets

We next describe the datasets we use. Our main source of Internet-wide data are frequent observations of ICMP latencies, extracted from publicly-provided Internet outage data from USC [97]. (The same source of latency data we use in Chapter 5.) We develop our algorithms with best-effort ground truth labeling of about 57k /24 blocks as fixed-line (wired), WiFi, or cellular. The labeled data includes about 36k fixed-line, 2.8k WiFi, and 18.3k cellular blocks.

We focus on networks *blocks* of 256-adjacent IPv4 addresses with the same /24 prefix. Prior work has suggested many such blocks are often used consistently [12, 130]. A more recent work has shown that 90% of the 1.97M /24 blocks they examined are homogeneous in terms of topological proximity [76].

6.2.1 Latency Information from USC Internet Outage Data

In this chapter we present a method to identify network type, with focus on cellular networks, by trends in their latency, so we require Internet-wide data that contains frequent latency estimates. While many groups today conduct censuses of the IPv4 address space, such scans are often infrequent or lack latency estimates. We extract latency estimates from publicly available measurements taken for Internet outage detection [98] using Trinocular [101]—we use another quarter (2017q3) from the dataset we describe in §5.2.1.

Trinocular collects data from six vantage points (VPs) positioned around the world. We use data from the Colorado vantage point—one of six sites— collected during September 2017 [98]. We find we get similar results regardless of the VP's site (§6.3.2).

Trinocular can only report latency when some address replies to a ping. We define a *latency-valid day*, for a /24 block, as one that has at least 10 observations. Most days of most blocks are latency-valid. For vantage point *c*, about 3.72M (98%) of the 3.78M blocks in the September 2017 dataset have at least 10 latency-valid days and about 96% of all blocks have 50 or more measurements on latency-valid days. We use the 3.72M blocks with 10 or more latency-valid days in the remainder of this paper.

6.2.2 Supporting Datasets

To provide context for the data we use reverse DNS, AS, and organization data. We use reverse DNS data from the Rapid7 dataset [102] taken September 2017, the same month as our latency data (§6.2.1).

We identify block users with AS routing and confirm organizations from CAIDA's Prefix-to-AS mappings dataset [19], derived from RouteViews *rv2* collector BGP snapshots data. The BGP

data is from September 14, 2017 (the same month as our latency data). Since some organizations use multiple ASes, we identify organizations using AS-to-organization mapping from the CAIDA AS Organizations Dataset [18].

6.2.3 Labeled Data (Best-Effort Ground Truth)

We evaluate our detection algorithms using *labeled data* as best-effort, ground-truth. We label each block as fixed-line, WiFi, or cellular. We call it *best-effort* because we use only *public* information—we are not able to verify each block with its operator. We describe how we identify each class below.

We use reverse DNS data (§6.2.2) to identify /24 blocks that are likely WiFi in our latency data (§6.2.1). We identify WiFi addresses as those with any of the keywords "wireless", "wifi", or "wlan" (case insensitive) in the reverse DNS name. Next we identify *WiFi-dominant* blocks, ones where the majority of DNS names (at least 128 of the 256 addresses in the /24 prefix) indicate WiFi connection. Being a WiFi-dominant /24 block, according to our definition, strongly indicates the block is in a WiFi network, but we use AS and organization data for further validation as we show next.

Using AS and organization data from CAIDA (as described in §6.2.2), we find 708 WiFi-dominant blocks in 29 universities. Since universities often identify WiFi networks with distinctive DNS names, we accept these blocks as part of our WiFi ground truth. Note that lowering the threshold to 64 WiFi addresses per block brings in only 13 additional university blocks, so we believe our threshold is reasonable. We limit our use of the keyword *wireless* to only blocks within universities to avoid false positives that can result from DNS names used by cellular operators. We assume that no universities operate cellular networks on public IP addresses because operation of cellular networks require government approval and licensing.

In the remaining WiFi-dominant blocks, we identify 1,333 blocks by the keyword *wifi*. We only kept 1206 blocks in 71 ASes for which we were able to verify, using information from their organizations' websites, that they strictly or partially provide Internet wireless services. We also

kept 841 blocks in 21 ASes out of 893 blocks identified by the keyword *wlan* using the same criteria. In summary, our WiFi ground truth contains 2,755 /24 blocks.

To identify fixed-line blocks, we begin by identifying 785 universities from around the world for which we have ping data. We use universities because many have large, public IP address spaces, and because they often identify WiFi networks with distinctive DNS names. We identify as fixed-line blocks all those belonging to known university prefixes that are not identified as WiFi. We identify about 36k /24 blocks as wired (non-WiFi) from 785 different universities around the world. We identify these blocks by both reverse DNS names and AS routing information as described in §6.2.2.

To find cellular blocks for our labeled dataset, we start with a list of 5 known wireless service carriers in different countries: AT&T Mobility (USA), Verizon (USA), Telia Company (Sweden), Vodacom (South Africa), and T-Mobile Polska (Poland). We identify the blocks' IP addresses association with the ASes and their organization using supplemental data (§6.2.2).

We use reverse DNS data to confirm that blocks associated with these carriers are designated for cellular services. For example, we find 6.2M reverse DNS records for AT&T mobility IP addresses with the format `mobile-x.mycingular.net` indicating AT&T mobility wireless address, whereas most reverse DNS names for the Verizon addresses follow the form `X.myvzw.com` indicating Verizon wireless address, where the X often reflects the IPv4 address. For example, the reverse DNS name for IP address 166.148.23.10 is `10.sub-166-148-23.myvzw.com`.

We find ping data for about 18.3k blocks in the USC ping dataset pertaining to the 5 carriers and have reverse DNS data that confirm they are cellular. For Verizon, the largest of the four major U.S. cellular network operators as of 2018q2, we identify about 10,786 /24 blocks, AT&T Mobility (565), Telia Company (1,173), Vodacom (5,659), and T-Mobile Polska (74). We use and label these blocks as our cellular ground truth data.

We believe our cellular ground data is reasonably representative. Our data comes from 5 different operators in different continents (North America, Europe, and Africa). Additionally, both,

fourth-generation (4G) and third-generation (3G) telecommunication systems are represented in our data. For example, the reverse DNS names of most of the Vodacom addresses contain the string `umts.vodacom.co.za` indicating 3G, while the majority of users in both U.S. carriers (Verizon and AT&T) had 4G availability around the time of our data collection [89].

We combine these three labeled datasets of 57k /24 blocks with our latency data to identify latency of known fixed-line (wired), WiFi, and cellular networks. We use 50% randomly selected blocks from each type as our training dataset to identify our classification algorithm thresholds, and the remaining half as the test dataset to evaluate the selected thresholds (§6.3).

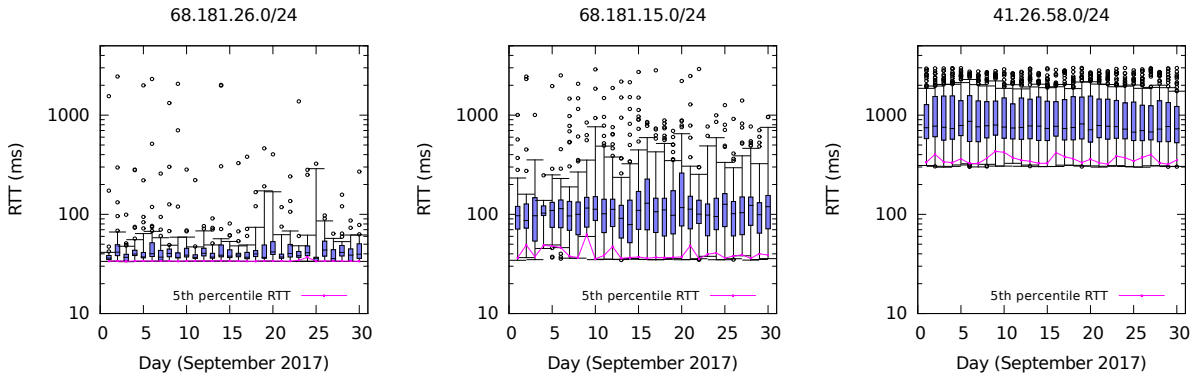
6.3 Identifying Block Type from Latency Estimates

We explain next our method to identify cellular blocks from latency measurements of /24 address blocks. We first show how latency can indicate network type for specific blocks in §6.3.1. We examine our labeled data from §6.2.3 to see the range of responses we observe around the Internet (§6.3.2). We then use ROC analysis to select proper thresholds for a classification algorithm (§6.3.3). Finally, we validate our thresholds settings using the test labeled data (§6.3.4).

6.3.1 RTT Variation in Known Blocks

We first look at three known blocks to consider how RTT varies when observed over one month of data. We use fixed-line and WiFi blocks from the University of Southern California, where we were able to confirm them as ground truth from discussion with their network operators. We select a cellular block from Vodacom, a wireless network operator in South Africa. We use this block because the AS name identifies a well known mobile network provider, and the reverse DNS names (e.g., `vc-gp-s-41-26-58-1.umts.vodacom.co.za.`) indicate a UTMS (3G) cellular network.

Each day of our latency data contains about 130 latency observations for each block. (There is no latency reported if there is no reply after 15 tries, so some days have fewer than 130 observations.) Figure 6.1 shows box plots for each of the 30 days of September 2017 for sample



(a) Fixed-line block. **(b)** WiFi block. **(c)** Cellular block.

Figure 6.1: Examples of daily RTT variability for /24 blocks with different Internet access types during September 2017. Variation in RTT (on *y* axis in log-scale) for cellular blocks is very different compared to fixed-line and WiFi blocks.

blocks of each type of network (fixed, WiFi, and cellular), measured from one vantage point in Colorado. Each box shows the interquartile, with observations from 25%ile to 75%ile, with a center line showing the median. The whiskers span 90% of RTTs. Circles show outliers defined as RTTs that lie outside the range of the whiskers. The magenta line shows the 5%ile value. Out of the 130 observations, this value is nearly always close to the minimum observed RTT over each day.

These three examples show that *daily RTT variation is an indicator of network type, especially cellular networks.* We can measure variation with the Interquartile Range (IQR), the difference between 25% and 75% observations over the course of a day. Both wireless blocks show higher daily RTT variation compared to the fixed-line block, and the cellular block has much larger IQR than the WiFi block. For these examples, the IQR of the fixed-line block ranges from about 2 ms to 15 ms, and from 21 ms to 175 ms for the WiFi block, and from 682 ms to 1026 ms for the cellular block. These patterns hold in the face of outliers, which often result due to transient congestion in the network. Moreover, since our measurements are taken over the course of 24 hours, this data is robust to medium-term congestion (perhaps due to popular events) and diurnal effects.

These examples suggest a method to identify cellular networks from only sparsely collected latency measurements. Next, we show that cellular networks consistently exhibit larger variance in latency compared to fixed-line and WiFi networks, and that since we measure *variation* in latency (and not absolute latency) this approach is robust to vantage point location.

6.3.2 Consistency of Variance in Latency in Each Block Type

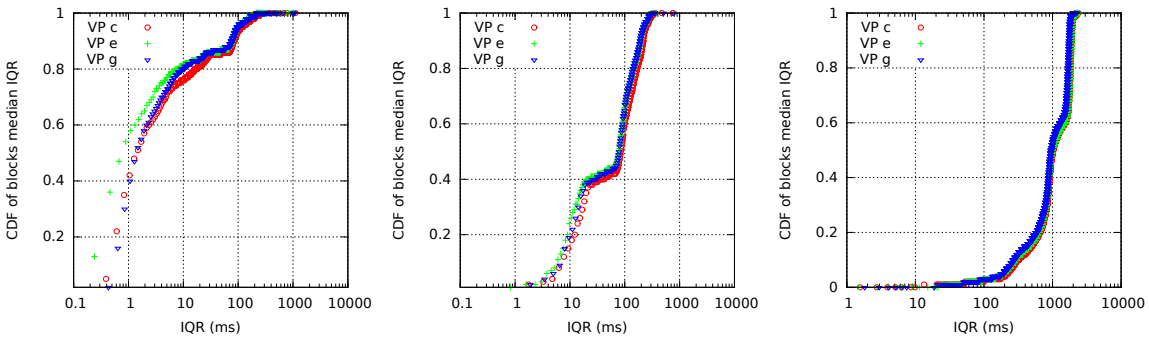
In §6.3.1, we show, using specific blocks, that variance in latency for a cellular block is distinctly different from fixed-line and WiFi blocks. We next generalize these examples to show that this trend can usually identify blocks in cellular networks.

In this section, we turn to our labeled data (§6.2.3) and examine how daily variation in RTT compares across the thousands of blocks from each type of network (fixed-line, WiFi, and cellular). Following §6.3.1, we use the interquartile range (IQR) of RTT over each day for each block, but here we distill observations for each block to one number: the median IQR for each block over the month, or just *median IQR*.

We define the *median IQR* of a /24 block as the median (middle-valued) IQR of all reported IQRs over one month (September 2017). For each block, we evaluate the IQR for each day with at least 10 latency observations, so each block has up to 30 daily IQRs over the month. (RTT cannot be observed if no IP address responds to an ICMP echo-request, so we cannot evaluate latency if the block is temporarily unused, or if there is a network outage.)

Figure 6.2 shows the distribution of the median IQR for randomly selected blocks from each block type in our best-effort ground-truth dataset (i.e., 50% of each type). We report observations from three of our six VPs in different locations (Colorado, c; U.S. east coast, e; and Greece, g).

First, we see that *variation in median IQR is a good predictor of cellular networks*, showing that the observations from Figure 6.1 generalize across thousands of blocks in labeled data. Known cellular networks (Figure 6.2c) show relatively large median IQRs: about 95.6% have median IQR above 250 ms, compared to only 0.4% of fixed and 4.5% of WiFi blocks.



(a) Fixed-line. **(b)** WiFi. **(c)** Cellular.

Figure 6.2: Distributions of median IQR for randomly selected blocks (i.e., 50% from each block category in our labeled datasets). Median IQR increases from fixed to WiFi to cellular blocks and is most distinctive in cellular blocks.

Second, the median IQRs of the fixed-line and WiFi blocks show significant overlapping. For example, 76% of fixed-line and 15.5% of WiFi blocks are below 10ms. That said, we also observe that blocks in WiFi networks show relatively higher variation in latency compared to those in fixed-line networks. For example, 36% of the WiFi blocks have median IQR of 110ms or higher compared to only 5% of fixed-line blocks. These results indicate we cannot distinguish between most of the WiFi and fixed-line blocks by the variation in their latency.

Third, we see that *VP location does not affect differences of network type.* While the absolute RTT can be very different from one VP to another, we see that RTT *variations* are very similar across all the 3 VPs on two different continents. (We see similar trends from the other three VPs.) Since the variation results from all VPs are consistent, we use only one of them in the remainder of this paper (i.e., VP *c*).

6.3.3 Methodology and Classification Thresholds

Our examples (§6.3.1) suggested we can use variation in RTT to identify blocks in cellular networks. An examination of large numbers of blocks (§6.3.2) showed that median IQR is distinct for the cellular networks, but has some overlap. The median IQR is also distinct for about one third of the labeled WiFi blocks that have relatively high variation in latency compared to the fixed-line blocks.

Table 6.1: Rules to classify blocks by their median IQR.

Rule	Prediction
median IQR < fwThresh	mixed (fixed-line or WiFi)
fwThresh ≤ median IQR < cellThresh	WiFi
median IQR ≥ cellThresh	cellular

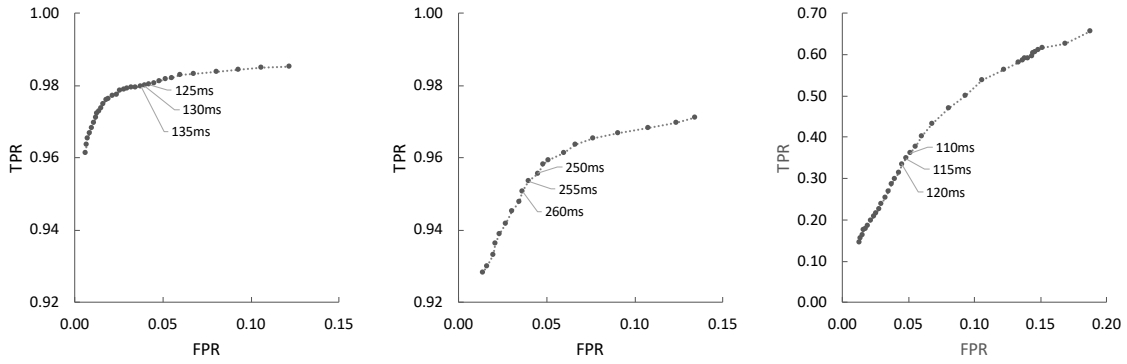
We suggest three root causes in cellular networks result in high IRQ. First, others have observed high wake-up times for wireless networks due to MAC-layer negotiation [90]. Second, prior work has observed large buffering (bufferbloat) in cellular networks can result in large delay and jitter [47, 69]. Third, Prior work has shown that the IP addresses in cellular networks are frequently shared across large geographic areas [11, 124, 131]. These configurations explain why cellular IQRs are larger than traditional networks.

We next use the labeled data (§6.2.3) to define a classification algorithm (Table 6.1) with two thresholds based on the median IQR: *cellThresh*, the threshold above which a network is identified as *cellular*, and *fwThresh*, the threshold below which a network is identified as *mixed* (i.e., fixed-line or WiFi), Networks with median IQR between these two thresholds are identified as *strictly WiFi*.

We randomly split our labeled data into two halves, one is for training and the other is for testing. We use the training part to learn the classification thresholds (this section), and the testing part to evaluate the learned thresholds (§6.3.4).

To select good thresholds, we consider this problem as a form of multi-class supervised learning. We then simplify our multi-class problem into three two-class classification problems. For each pair of classes, we use *receiver operating characteristic* (ROC) curves to select the thresholds that yield the best true positive rate (TPR) *vs.* false positive rate (FPR) results computed using the confusion matrices. Our main goal is to detect cellular blocks with high accuracy, but we also find a median IQR range that identifies part of the WiFi blocks.

First, we determine a cellThresh that would identify cellular blocks using ROC curve for the results of cellular *vs.* fixed-line binary classification (Figure 6.3a) and cellular *vs.* WiFi (Figure 6.3b) at different cellThresh values. Figure 6.3a shows that the cellular and fixed-line blocks



(a) Cellular *vs.* fixed-line. **(b)** Cellular *vs.* WiFi. **(c)** WiFi *vs.* fixed-line.

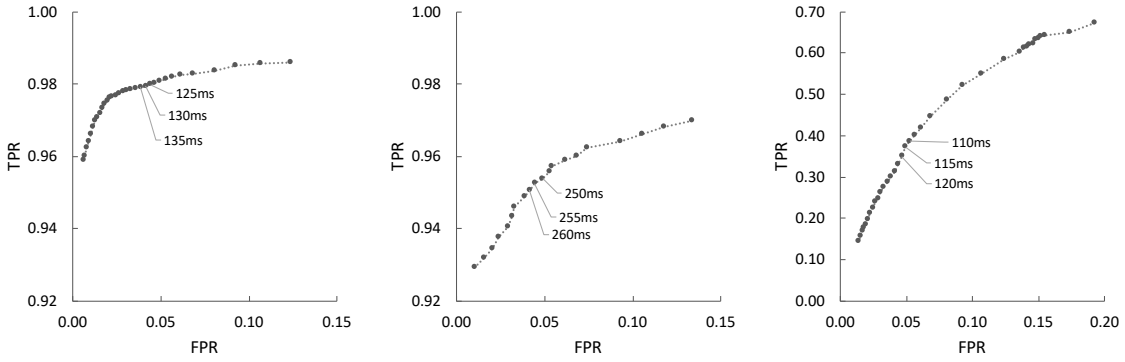
Figure 6.3: ROC curves to show the TPR *vs.* FPR of the classification results over the labeled training data at different cellThresh (Figure 6.3a and Figure 6.3b) and fwThresh (Figure 6.3c) values shown next to some of the marks.

are easy to separate. A cellThresh of 130 ms gives a TPR of 98% and a FPR of only 4%. Figure 6.3b shows that it is also easy to separate cellular and WiFi blocks but at higher thresholds. A cellThresh of 255 ms would still detect most of the cellular blocks in the training data with a TPR of 95.4% and a low FPR of 4% (i.e., WiFi blocks identified as cellular). We conclude that a cellThresh of 255 ms is good to identify most of the cellular blocks with low FPR for both fixed-line and WiFi types.

Next, we determine the median IQR overlapping range for fixed-line and WiFi blocks. Our goal is to determine *fwThresh* according to Table 6.1. Figure 6.3c shows the ROC curve for WiFi *vs.* fixed-line classification results at different fwThresh values. We vary fwThresh from 20 ms and up to 200 ms (5 ms increments). We see significant overlapping up to 115 ms, from there, 35% of the WiFi blocks have higher median IQRs compared to only 4.8% of the fixed-line. Smaller fwThresh values would identify more WiFi blocks but with FPR greater than 5%. We set fwThresh to 115 ms, below which blocks are identified as *mixed*. Blocks with median IQR between fwThresh and cellThresh (i.e., 255 ms) are classified as *strictly WiFi*.

6.3.4 Evaluation with the Test Labeled Data

We evaluate our selection of thresholds (§6.3.3) on the withheld test labeled dataset.We find that the distributions of median IQR (Figure 6.2) show only minor changes when reevaluated



(a) Cellular *vs.* fixed-line. **(b)** Cellular *vs.* WiFi. **(c)** WiFi *vs.* fixed-line

Figure 6.4: ROC curves of classification results over the test data confirm our findings over the training data in Figure 6.3a, Figure 6.3b, and Figure 6.3c.

with the test subset (we do not show here due to space). The ROC curves on the withheld evaluation subset depicted in Figure 6.4 are also similar to those in Figure 6.3a, Figure 6.3b, and Figure 6.3c.

The classification results over the test data confirms that our thresholds settings are good for the test data and give similar results to those over the training data. These result are not surprising because our approach is relatively stable to a range of parameters (as shown in §6.3.3) and our goal is to select *good* parameters rather than strictly *optimal*. We report the classification accuracy results in §6.5.1.

6.4 Identifying Block Types in the Wild

We next apply our method to Internet-wide data (§6.2.1). We measure the variation in latency for each block by the median IQR (§6.3.2). We then apply our classification algorithm (Table 6.1) with the parameters fwThresh set to 115 ms and cellThresh set to 255 ms (§6.3.3).

Table 6.2 summarizes the classification results. The table shows for each class the number and fraction of blocks we found, as well as the number of unique ASes and organizations that have some of those blocks. (Organizations and ASes may be counted multiple times if they have blocks of different types.) We derive the information for blocks ASes and organizations using our supplemental AS and organization data (§6.2.2). Our classification identify a significant

Table 6.2: Summary of classification results.

Class	/24 blocks	% of total	#ASes	#organizations
Mixed	3,524k	94.83%	40.6k	37.4k
WiFi	23k	0.62%	1.5k	1.4k
Cellular	169k	4.55%	1.1k	1k
all	3,716k	100%		

Table 6.3: Top 5 organization with blocks identified as mixed.

Organization	/24 blocks	ASes (3 at most)
CHINANET-BACKBONE	189.5k	4134
Comcast Cable Communications, LLC	151.6k	7922, 7015, 33491
CNCGROUP Jitong IP network	140.5k	4837, 9929
Korea Telecom	118.2k	4766, 45400
Deutsche Telekom AG	82.1k	3320, 2792, 8204

fraction of the address space as mixed, about 3.5M blocks (94.8%). We identify about 23k (0.6%) blocks as WiFi, and 169k (4.6%) blocks as cellular.

To understand these results, we identify the top organizations in each classification category. Table 6.3 list the top 5 organizations sorted by the number of blocks identified as mixed, and up to three ASes sorted by the number of blocks we identify for each. As one would expect, these organizations are all ISPs with a presence in many homes ("eyeball" networks).

Similarly, Table 6.4 lists the top 5 organizations for blocks identified as strictly WiFi. Our algorithm identifies only 23k (0.6%) of the 3.72M blocks in the USC ping dataset as WiFi. This result is not surprising as we expect that most blocks with WiFi access would fall in the *mixed* category as the results discussed in §6.3.3 suggest.

We classify several satellite ISPs as showing large IQRs (but not as large as cellular networks). We did not have satellite ISPs in our training data, but the results of classifying blocks from several satellite ISPs suggest they will not be confused with cellular. We examined how our method classified blocks from 9 satellite ISPs—studied in [90]— including Hughes Network (ranked 3rd) and ViaSat (4th) in Table 6.4. We find 10.37k blocks from the 9 ISPs in our USC ping dataset—96.5% of them belong to ViaSat, Hughes, and iiNet. We identify 82.4% of the 10.37k blocks as mixed, 13.5% as strictly WiFi, and only 4.1% as cellular. The results suggest satellite access has

Table 6.4: Top 5 organization with blocks identified as strictly WiFi.

Organization	/24 blocks	ASes
CHINANET-BACKBONE	3.58k	4134
CANTV Servicios, Venezuela	1.00k	8048
Hughes Network Systems	0.64k	4812
ViaSat,Inc.	0.63k	7155
Akamai International B.V.	0.57k	20940

Table 6.5: Top 10 organization with blocks identified as cellular.

Organization	/24 blocks	ASes
Tim Celular S.A.	32k	26615
TELEFÔNICA BRASIL S.A	29.4k	26599
Itissalat Al-MAGHRIB	14.6k	6713
Cellco Partnership DBA Verizon Wireless	10.1k	22394, 6167
TELE2	6.9k	1257
Vodacom	6.2k	29975, 36994
NTT Communications Corp.	4.7k	4713
Telkom SA Ltd.	3.2k	37457, 5713
NTT DOCOMO, INC.	2.8k	9605
Hi3G Access AB	2.7k	44034

a distribution of latency as WiFi and fixed-line, and its effect on identifying cellular blocks is insignificant.

Finally, Table 6.5 shows the top 10 organizations of blocks identified as cellular. We note that all of these organization are well known cellular service providers from around the world. Together, these organizations contribute around 112.6k (66.6%) of all the blocks identified as cellular. Note that the absolute number of cellular blocks seems small relative to the number of mobile users (more than 5 billion as of 2017). Most mobile phones use network-address translation (NAT) and so do not have public IP address most of the time. Prior work has shown that cellular traffic is concentrated in a small number of /24 blocks attributing that to carrier-grade NATs (CGNs) presence in cellular networks [109].

Table 6.6: Labeled test dataset blocks classification results.

Block type	#Blocks	Predictions		
		Mixed	Strictly WiFi	Cellular
Fixed-line	17,980	17,094	816	70
WiFi	1,378	864	453	61
Cellular	9,129	175	258	8,696

6.5 Validation

We already evaluated our block type identification algorithm over the labeled test dataset of 28.5K blocks and showed that the classification results are similar to those over the training dataset (§6.3.4). In §6.5.1, we discuss the accuracy of the classification results with the selected classification parameters over the test dataset. We also confirm that results are robust to different variance metrics in §6.5.2.

6.5.1 Classification Accuracy

Table 6.6 shows our classification *predictions* for the labeled blocks in the test dataset, while Table 6.7 shows the TPR and FPR for these results. The results in Table 6.6 demonstrate our algorithm ability to identify most of the cellular blocks with low number of false positives (70 fixed-line and 61 WiFi). We correctly classify 8,696 (95.3%) of the blocks labeled cellular.

The results over the fixed-line and WiFi test data blocks are consistent with the results over the training data (§6.3.3) where we expect most of them to fall in the *mixed* class. Our method correctly identify 95.1% of fixed-line blocks as mixed and the majority of the WiFi blocks (95.6%) as mixed (62.7%) or strictly WiFi (32.9%).

Table 6.7 reports the TPR and FPR of the results in Table 6.6. To show our method's ability to identify cellular blocks, we compute the TPR and FPR for two categories: (1) *fixed-line and WiFi* (i.e., all 19,358 blocks labeled fixed-line or WiFi in the test set), (2) *cellular* (i.e., 9,129 blocks labeled cellular). For the *fixed-line and WiFi* category, we count as true positives (TPs) the blocks labeled fixed-line we identify as mixed, and the WiFi blocks we identify as mixed or strictly WiFi. We count as false negatives (FNs) the fixed-line blocks we identify as strictly WiFi

Table 6.7: TPR and FPR for the classification of the test data.

Block type	Blocks	TPR	FPR
Fixed-line and WiFi	19,358	95.1%	4.7%
Cellular	9,129	95.3%	0.7%

or cellular, and the WiFi blocks we identify as cellular. For the *cellular* category, we count the cellular blocks we identify as cellular as TPs, and as FNs otherwise. From the table we see that our method accurately identifies cellular blocks with a TPR of 95.3% and a very low FPR of 0.7%.

Our results are correct (over 95% of ground truth cellular is correctly identified), but we do not claim the results are complete. We cannot classify blocks for which we have no data, and we find many blocks of our five carriers do not appear in ping observations because the blocks are not responsive. (We have data for only about 18.3k blocks, one-third of the 53k assigned to the carriers and with DNS names indicating cellular use (§6.2.2).) These blocks may be currently unused, or they may be firewalled.

6.5.2 Sensitivity of Classification to IQR

Given the classification results in the wild, how often would we get a different answer due to variation in IQR. To show how sensitive is our classification model to the variation in RTT, we first compute for each block in each class, the *median* IQR and the first and third quartiles of all daily IQRs during September 2017. We then plot the results against our classification thresholds as Figure 6.5 shows.

Figure 6.5 depicts the median IQR (solid lines) and the IQR variation (filled curves) for sample blocks identified by our method as mixed, strictly WiFi, and cellular. To avoid slow rendering, we downsample the blocks before plotting, taking every fifth cellular, and every 100th mixed, after sorting by the median IQR. The results are visually the same as the complete data. We show our fwThresh and cellThresh on the graph indicated by the dashed horizontal lines.

These results show that our ROC analysis of labeled data (§6.3.3) applies to the whole dataset. The daily IQRs of the blocks identified as cellular show relatively large variation throughout

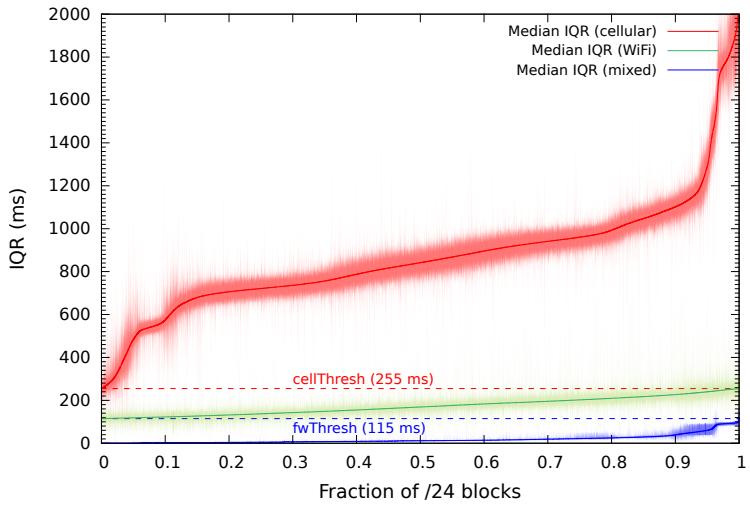


Figure 6.5: Median IQR and the daily IQRs variation by block type during September 2017. Blocks identified as cellular show high variation in daily IQRs but are consistently higher than cellThresh.

September, but are consistently above cellThresh. The daily IQRs of blocks identified as WiFi and mixed show less variation and are consistently below cellThresh.

6.6 Conclusions

This chapter defined a new method to identify a cellular /24 IP block by variance in its latency measurements (§6.3.3). We measured the daily variance in a block's latency using the *interquartile range* of RTTs in a day and then used the *median IQR* of all daily IQRs over a month to distinguish block type. We showed that the variation patterns are consistent over time for a block. As a result, we do not need such an extended period to observe the variation trend of a block's latency measurements. We found that the median IQRs of fixed-line and WiFi blocks largely overlap, allowing only a fraction of the WiFi blocks to be identified with high confidence. As a result, we identified three categories (mixed, WiFi, cellular) of block type with two thresholds of median IQR, and selected good thresholds based on our labeled data, a form of best-effort ground-truth.

We applied our method to 3.72M /24 blocks and found that the majority of them were identified as mixed (fixed-line and WiFi) (94.8%), 0.6% as WiFi, and 4.6% as cellular. We validated part of our algorithm predictions, showing high accuracy in cellular blocks identification (§6.5).

This chapter assists the delay-based methods we propose in this thesis for characterizing IP addresses co-locality and movement become more accurate by defining a method to automatically identify cellular blocks, a category that presents a challenge to delay-based methods. The high variation in latency of cellular blocks compared to other types suggests that cellular blocks present a potential challenge to methods that rely on latency measurement, such as those we propose in this thesis (Chapter 4 and Chapter 5), and many measurement-based geolocation methods (§2.3). Our client-independent method for identifying cellular blocks serves as a step toward studying the impact of these blocks on delay-based methods, providing an opportunity to improve their accuracy.

Chapter 7

Future Work and Conclusions

In this chapter, we present possible directions for future work and conclude this thesis.

7.1 Future Work

There are several directions for immediate future work that could strengthen, expand, and enrich our studies. We next discuss possible future work related to our four studies in this dissertation.

In Chapter 3, we evaluated router geolocation in four public and commercial geolocation services. There are several directions for future work that can enrich this work. First, this evaluation can be extended to include the geolocation of additional network infrastructure, such as HTTP servers, DNS servers, and Content Delivery Networks (CDNs) Front-Ends. The work can be extend furthermore by including more popular services. Given the Internet dynamics and that geolocation service providers continuously update their services, it is beneficial to reevaluate them from time to time. Such evaluation would raise awareness about the limitations the services may have and perhaps push providers to address them. Second, we hypothesized about some of the techniques that geolocation service providers might have used to compile their IP to location mappings. Future work can dig deeper to identify what methods different providers use to collect location information and identify sources that lead to incorrect geolocation. Finally, a related future work could identify blocks that contain IP addresses of network infrastructure and study their geographic distribution and track their movement.

In Chapter 4, we devised a delay-based method to assess the co-locality of IP addresses. Part of the study assessed the frequent assumption that IP addresses in a /24 block are proximate. There are several future work directions to strengthen this study. First, given the depletion of the IPv4 address space, it would be interesting to assess the trends in co-locality of /24 blocks. We think there may be even more blocks with endpoints at different locations as ISPs look to

address the scarcity of IPv4 addresses with better address space utilization. Second, our study did not consider the effect that cellular blocks might have on the results. In Chapter 6, we estimated that around 4.6% of 3.72M responsive /24 blocks are cellular. We observed large variation in the latency of these blocks, which could lead our clustering algorithm to identify their IP addresses as un-clustered or not-co-located. Future work can quantify cellular blocks identified as multi-location blocks by our co-locality assessment method from Chapter 4. This future work direction should also study the geographic distribution of addresses in cellular blocks, which may indeed be at very different locations [131]. Note that the dataset in §4.2 is not suitable for our algorithm to identify cellular blocks. (The cellular identification algorithm requires several observations throughout the day and over several days, while the dataset in §4.2 reports only the minimum of 10 measurements to an IP address taken over a short time.) Finally, our delay-based clustering method can be part of a long-lived system that tracks the stability of IP addresses co-locality over time. Such a long-lived system can also support a geolocation service by identifying ranges of IP addresses mapped to the same location when they appear to have endpoints at different locations.

In Chapter 5, we devised a delay-based algorithm to identify block movement. We applied our algorithm to identify block movement in two quarters worth of raw probing data, 2018q4 and 2019q1. An immediate future work direction is to expand the study to include more quarters to study trends in block movement over time. This longitudinal study can help quantify the rate at which blocks move, allowing for identifying which blocks are location-stable and which are more location-dynamic over time. In §5.5.3 we identified real-world block movement examples from inter-RIR transfer logs and used them to validate our work. However, most transfers appear to involve blocks that were unused previously. As a result, we only found a few blocks with valid ping data before and after being transferred. As future work, we want to identify more real-world block movement to solidify our validation.

In Chapter 6, we implemented a method to identify cellular blocks by their distinctive variation in RTT as we probe them over time, without proprietary data from network providers.

There are several future work directions for a better understanding of this important class of blocks. First, Prior work, around 10 years old studies, showed that cellular networks' IP addresses are frequently shared across many distinct locations [11, 131]. In that direction, future work can study how different carriers manage their cellular address blocks concerning their geographic distribution and movement and show if the results of these old studies still hold. Second, we make the assumption, based on prior work, that cellular /24 blocks are homogeneous, meaning it is unlikely that the cellular blocks have IP addresses with mixed access types, cellular and non-cellular. Future work can confirm if this assumption is valid by studying the RTT variation of individual addresses in blocks identified as cellular. We expect most of the addresses to have high IQRs if the assumption is valid.

Besides the above immediate future work closely related to our four studies, our thesis suggests a new scope for these studies. These studies only investigated the IPv4 address space. As the deployment of IPv6 grows [27, 104], future work can explore the applicability of our proposed delay-based methods (to assess IP addresses co-locality (Chapter 4), detect block movement (Chapter 5), and identify cellular blocks (Chapter 6)) to the IPv6 address space. Given how large the IPv6 address space size is (2^{128}), we expect different challenges with respect to determining visibility and collecting measurement to the IPv6 addresses [86].

7.2 Conclusions

More and more network research, as well as Internet service providers and their clients, benefit from IP geolocation. Occasionally, IP address blocks get traded or reassigned or become available after being unused, suggesting that we need to implement IP geolocation as a continuous service to keep the location of all IP addresses up-to-date. However, frequent geolocation of all IP addresses is inefficient and unnecessary.

This dissertation hypothesized that *we can enable efficient, continuous geolocation by identifying clusters of co-located IP addresses and their location stability from latency observations.* We started by motivating the thesis statement with an evaluation study of router geolocation

in popular geolocation services (Chapter 3). To demonstrate this thesis statement, we presented new optimizations that leverage delay measurement to identify arbitrary-size groups of co-located addresses (Chapter 4) and to detect when an address block moves (Chapter 5). These optimizations can be used by a geolocation system to identify which groups of addresses can be geolocated as a unit, as well as when a geolocation update is needed.

In Chapter 3, we motivated our thesis statement by presenting an evaluation study of router geolocation in widely-used, public and commercial geolocation services (provided as IP-to-location databases). We evaluated the accuracy of the databases with a ground truth datasets we created using DNS-based and latency-based methods. We showed that these services can be inaccurate for router geolocation at both country- and city-level resolutions, corroborating previous evaluation studies on the overall reliability of geolocation services (§2.4). Furthermore, we show that some of these databases lack extensive city-level coverage (§3.5.1). A breakdown of the ground truth by RIR showed that the databases are less reliable at the city-level resolution at ARIN region compared to the other regions. Furthermore, using validated heuristics to identify locations from DNS names, we showed that even IP addresses assigned to routers may experience location change over time (§3.3.4.2). We concluded that researchers need to pay extra caution if they need to use these geolocation databases and should investigate the impact they may have on their results. Our findings in this chapter motivate our work to enable more reliable geolocation that maintains up-to-date location mappings.

Our study in Chapter 4 demonstrates the part of the thesis statement that we can leverage latency observation to identify clusters of co-located IP addresses that we can treat as units to enable efficient geolocation (§4.6). In this chapter, we devised a delay-based method to identify clusters of adjacent, co-located IP addresses automatically. The method employs hierarchical clustering that groups together similar IP addresses in a multidimensional space of delay coordinates. We validated the method using single- and multi-location block datasets. We then used the method to assess the common assumption that IP addresses in a /24 block are co-located. Applying our method to 1.41M /24 blocks (118M addresses), we found that a noticeable frac-

tion of these blocks (17%) appear to have endpoints at different locations, with an estimated upper-bound false-positives rate of 5.4%. This outcome disagrees with the /24 block co-locality assumption. We then showed that the clustering method could identify larger groups of co-located IP addresses. We applied the method, individually, to 65 university groups, each with a different number of co-located /24 locks, 2 to 144, and a total of 1,974 blocks in all groups. The method correctly identified 56 (86%) of the groups as one cluster of IP addresses each. Moreover, the majority of the /24 blocks, 1,957 (99.1%), were identified as part of the majority cluster of each university. Consistently co-located groups can be treated as a unit, leading to a significant reduction in the number of targets to geolocate and consequently reducing the measurement load.

Our study in Chapter 5 demonstrates the part of the thesis statement that we can characterize the location stability of IP address blocks using latency observations from a handful of vantage points. This chapter defined and implemented a delay-based algorithm to identify if a block has moved by observing persistent, significant changes in the block's latency from multiple sites around the same time. Using our algorithm over an Internet-wide latency data §5.2.1, we showed that most blocks, as expected, appear to be stationary. We estimated that around 78.7k /24 blocks (2.1% of blocks with valid data) have moved during 2018q4 and around 65k blocks (1.7%) during 2019q1. These results show that only a small fraction of the IP blocks require geolocation updates, and our method allows a geolocation system to identify these blocks. Our method can be used to quickly identify potential IP block movement to help a geolocation service keep up-to-date geolocation. Moreover, using our method to study block movement over time allows for identifying location-stable and location-dynamic IP blocks, thus dictating the frequency at which different blocks require re-geolocation.

In Chapter 6, we extended our utilization of delay measurement to identify blocks in cellular networks, an interesting class of blocks for IP geolocation and many other network applications. Previous work has shown that geolocating addresses in cellular blocks can be challenging (§2.6). Our method to identify these cellular blocks is a step toward further future work to

study their geographic properties and impact on delay-based methods, such as those proposed in this thesis. In this chapter, we implemented a new delay-based method to classify blocks as *cellular*, *strictly WiFi*, or *mixed* (wired or WiFi) merely based on blocks' daily RTT patterns. We measured the daily variance in a block's latency using the *interquartile range* of RTTs in a day and then used the *median IQR* of all daily IQRs over a month to distinguish block type. Our method is most successful at identifying cellular blocks since their RTT patterns are the most distinguished. We applied this method to most of the public Internet, reporting on about 3.72M responsive IPv4 /24 blocks data from September 2017. We identify about 169k blocks (4.6%) as cellular, 23k (0.6%) as strictly WiFi, and a majority of 3.5M (94.8%) as mixed. We validated part of our algorithm predictions in the wild, showing high accuracy in cellular blocks identification. Our method can assist future studies that characterize how cellular networking affects the Internet without proprietary information from network providers.

To sum up, in this dissertation, we have shown we can leverage latency measurement from a small set of vantage points to characterize the IP addresses in ways useful to IP geolocation and other network applications. A geolocation system can integrate the proposed delay-based methods in this dissertation to maintain efficient, up-to-date IP-to-location service. Identifying co-located groups of IP addresses and the location dynamics of IP blocks will lead to a smarter measurement collection approach. We only need to measure and geolocate a few representatives from a consistently co-located collection of addresses. Moreover, we do not need to measure the stable-location IP addresses as frequently as the dynamic-location ones.

Bibliography

[1] The Google maps geocoding API. https://developers.google.com/maps/documentation/geocoding/, August 2015.

[2] Universities worldwide. http://univ.cc, July 2015.

[3] Network information api. https://wicg.github.io/netinfo/, 2018.

[4] Abdelrahman Abdou and P. C. Van Oorschot. Server location verification (slv) and server location pinning: Augmenting TLS authentication. *ACM Trans. Priv. Secur.*, 21(1):1:1–1:26, December 2017.

[5] Akamai. Akamai EdgeScape. https://developer.akamai.com/edgescape, December 2019.

[6] Tom Anderson, Ratul Mahajan, Neil Spring, and David Wetherall. Rocketfuel: An isp topology mapping engine. http://www.cs.washington.edu/research/networking/rocketfuel/, June 2013.

[7] APNIC. APNIC resource transfer log. ftp://ftp.apnic.net/public/transfers/apnic/, December 2019.

[8] M. J. Arif, S. Karunasekera, S. Kulkarni, A. Gunatilaka, and B. Ristic. Internet host geolocation using maximum likelihood estimation technique. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 422–429, 2010.

[9] ARIN. Statistics: Specified transfers of Internet number resources. https://account.arin.net/public/transfer-log, 2019.

[10] ARIN. Transferring IP addresses & ASNs. https://www.arin.net/resources/registry/transfers/, October 2019.

[11] Mahesh Balakrishnan, Iqbal Mohomed, and Venugopalan Ramasubramanian. Where's that phone? geolocating IP addresses on 3g networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, IMC '09, page 294–300, New York, NY, USA, 2009. Association for Computing Machinery.

[12] Xue Cai and John Heidemann. Understanding block-level address usage in the visible internet. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 99–110, NY, USA, 2010. ACM.

[13] CAIDA. The CAIDA UCSD Internet topology data kit - march 2018. `http://www.caida.org/data/internet-topology-data-kit`.

[14] CAIDA. The CAIDA UCSD IPv4 routed /24 topology dataset - october-december 2018. `https://www.impactcybertrust.org/`.

[15] CAIDA. AS rank. `http://as-rank.caida.org`, 2016.

[16] CAIDA. CAIDA topology dataset. `https://topo-data.caida.org/`, March 2016.

[17] CAIDA. Internet mapping and annotation. `http://www.caida.org/research/topology/internet_mapping/`, 2016.

[18] CAIDA. The CAIDA AS organizations dataset, 20171001. `http://www.caida.org/data/as-organizations`, October 2017.

[19] CAIDA. Routeviews Prefix-to-AS mappings (pfx2as) for IPv4 and IPv6, rv2-20170914-0200. `http://data.caida.org/datasets/routing/routeviews-prefix2as/`, September 2017.

[20] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. Mapping the expansion of google's serving infrastructure. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 313–326, New York, NY, USA, 2013. ACM.

[21] Balakrishnan Chandrasekaran, Mingru Bai, Michael Schoenfield, Arthur Berger, Nicole Theresa Caruso, George Economou, Stephen Gilliss, Bruce M. Maggs, Kyle Moses, David Duff, Keung-Chi Ng, Emin Gün Sirer, Richard Weber, and Bernard Wong. Alidade: IP geolocation without active probing. Technical Report CS-TR2015.001, Department of Computer Science, Duke University, 2015.

[22] Fangfei Chen, Ramesh K. Sitaraman, and Marcelo Torres. End-user mapping: Next generation request routing for content delivery. In *The ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 167–181, New York, NY, USA, 2015. ACM.

[23] Tsuwei Chen. An overview of Google location services. `https://www.netcheif.com/blog/wp-content/uploads/2011/11/mobile_location.pdf`, 2011.

[24] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. PlanetLab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003.

[25] CIA. The World Factbook. `https://www.cia.gov/library/publications/the-world-factbook/fields/2011.html`, 2017.

[26] Cisco. Cisco visual networking index: Forecast and methodology, 2016-2021. Technical report, June 2017.

[27] Jakub Czyz, Mark Allman, Jing Zhang, Scott Iekel-Johnson, Eric Osterweil, and Michael Bailey. Measuring IPv6 adoption. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, page 87–98, New York, NY, USA, 2014. Association for Computing Machinery.

[28] Alberto Dainotti, Karyn Benson, Alistair King, Bradley Huffaker, Eduard Glatz, Xenofontas Dimitropoulos, Philipp Richter, Alessandro Finamore, and Alex C. Snoeren. Lost in space:

Improving inference of IPv4 address space utilization. *IEEE Journal on Selected Areas in Communications*, 34(6):1862–1876, June 2016.

[29] C. Davis, P. Vixie, T. Goodwin, and I. Dickinson. A means for expressing location information in the domain name system. *RFC 1876*, 1996.

[30] DB-IP. The DB-IP databases. `https://db-ip.com`, 2017.

[31] Digital Element and Location Based Marketing Association. Digital data exhaust report 2018. `https://www.digitalelement.com/wp-content/uploads/2018/10/Digital-Data-Exhaust-Survey-Report-2018.pdf`, 2018.

[32] Digital Envoy. Digital Element NetAcuity databases. `https://www.digitalelement.com/netacuity/`, 2016.

[33] Digital TV Research. SVoD databook. `https://www.digitaltvresearch.com/ugc/press/254.pdf`, September 2019.

[34] Digital.gov. 2016/2017 mobile analysis: Mobile device trends on government websites. `https://digital.gov/2017/08/14/20162017-mobile-analysis-mobile-device-trends-on-government-websites`, August 2017.

[35] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide scanning and its security applications. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 605–620, Washington, D.C., 2013. USENIX.

[36] Ahmed Elmokashfi, Amund Kvalbein, Jie Xiang, and Kristian R. Evensen. Characterizing delays in Norwegian 3G networks. In Nina Taft and Fabio Ricciato, editors, *Passive and Active Measurement*, pages 136–146, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[37] Ericsson. Ericsson mobility report: June 2018. Technical report, June 2018.

[38] Brian Eriksson, Paul Barford, Bruce Maggs, and Robert Nowak. Posit: a lightweight approach for IP geolocation. *SIGMETRICS Perform. Eval. Rev.*, 40(2):2–11, October 2012.

[39] Brian Eriksson and Mark Crovella. Understanding geolocation accuracy using network geometry. In *Proceedings of the Infocom 2013 Miniconference*, Turin, Italy, 2013.

[40] European Commission. A digital single market strategy for europe. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:52015DC0192`, 2015.

[41] Xun Fan and John Heidemann. Selecting representative IP addresses for internet topology studies. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 411–423, New York, NY, USA, 2010. ACM.

[42] Xun Fan, Ethan Katz-Bassett, and John Heidemann. Assessing affinity between users and CDN sites. In *Workshop on Traffic Monitoring and Analysis (TMA)*, 2015.

[43] Michael J. Freedman, Mythili Vutukuru, Nick Feamster, and Hari Balakrishnan. Geographic locality of IP prefixes. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, IMC '05, pages 153–158, Berkeley, CA, USA, 2005. USENIX Association.

[44] Futuresource Consulting Ltd. SVoD market research, analysis and commentary. `https://www.futuresource-consulting.com/press-release/media-entertainment-press/new-services-set-to-drive-svod-revenues-up-25-to-usd-36-billion-in-2019/`, April 2019.

[45] GeoNames. The GeoNames Geographical Database. `http://www.geonames.org/`, January 2017.

[46] GeoNames. The GeoNames Geographical Database. `http://www.geonames.org/`, September 2019.

[47] Jim Gettys. Bufferbloat: Dark buffers in the internet. *IEEE Internet Computing*, 15(3), May 2011.

[48] Manaf Gharaibeh, Anant Shah, Bradley Huffaker, Han Zhang, Roya Ensafi, and Christos Papadopoulos. A look at router geolocation in public and commercial databases. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, pages 463–469, New York, NY, USA, 2017. ACM.

[49] Manaf Gharaibeh, Han Zhang, Christos Papadopoulos, and John Heidemann. Assessing co-locality of IP blocks. In *Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference*, pages 503–508. IEEE, 2016.

[50] Vasileios Giotsas, Amogh Dhamdhere, and Kimberly C. Claffy. Periscope: Unifying looking glass querying. In *Passive and Active Measurement - 17th International Conference, PAM 2016, Heraklion, Greece, March 31 - April 1, 2016. Proceedings*, pages 177–189, 2016.

[51] Vasileios Giotsas, Petros Gigis, Alexandros Milolidakis, Eric Nguyen Duy, Marios Isaakidis, and Edwards. Mukasa. The remote peering jedi a portal in the remote peering ecosystem. RIPE 73, October 2016.

[52] Mark Gondree and Zachary N.J. Peterson. Geolocation of data in the cloud. In *Proceedings of the third ACM conference on Data and application security and privacy*, CODASPY '13, pages 25–36, New York, NY, USA, 2013. ACM.

[53] Google. Google geolocation api. `https://developers.google.com/maps/documentation/geolocation/intro`, 2019.

[54] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for Internet map discovery. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, 2000.

[55] Bamba Gueye, Steve Uhlig, and Serge Fdida. Investigating the imprecision of IP block-based geolocation. In *Proceedings of the 8th International Conference on Passive and Active Network Measurement*, PAM'07, pages 237–240, Berlin, Heidelberg, 2007. Springer-Verlag.

[56] Bamba Gueye, Artur Ziviani, Mark Crovella, and Serge Fdida. Constraint-based geolocation of Internet hosts. *IEEE/ACM Trans. Netw.*, 14(6):1219–1232, December 2006.

[57] Krishna P. Gummadi, Stefan Saroiu, and Steven D. Gribble. King: estimating latency between arbitrary Internet end hosts. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, IMW '02, pages 5–18, New York, NY, USA, 2002. ACM.

[58] Chuanxiong Guo, Yunxin Liu, Wenchao Shen, Helen J. Wang, Qing Yu, and Yongguang Zhang. Mining the web and the Internet for accurate IP address geolocations. In *INFOCOM*, pages 2841–2845, 2009.

[59] John Heidemann, Yuri Pradkin, and Guillermo Baltra. The policy potential of measuring Internet outages. In *Proceedings of the TPRC, the Research Conference on Communication, Information and Internet Policy*, Washington, DC, USA, September 2018. pub-tprc.

[60] John Heidemann, Yuri Pradkin, Ramesh Govindan, Christos Papadopoulos, Genevieve Bartlett, and Joseph Bannister. Census and survey of the visible Internet. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 169–182, New York, NY, USA, 2008. ACM.

[61] Zi Hu, John Heidemann, and Yuri Pradkin. Towards geolocation of millions of IP addresses. In *The 2012 ACM conference on Internet measurement conference*, IMC '12, pages 123–130, New York, NY, USA, 2012. ACM.

[62] B. Huffaker, M. Fomenkov, and k. claffy. Geocompare: a comparison of public and commercial geolocation databases - Technical Report . Technical report, Cooperative Association for Internet Data Analysis (CAIDA), May 2011.

[63] Bradley Huffaker, Marina Fomenkov, and kc claffy. Geocompare: a Comparison of Public and Commercial Geolocation Databases. Technical report, Cooperative Association for Internet Data Analysis (CAIDA), May 2011.

[64] Bradley Huffaker, Marina Fomenkov, and kc claffy. DRoP: DNS-based router positioning. *ACM SIGCOMM Computer Communication Review*, 44(3):5–13, 2014.

[65] IANA. Regional internet registry (RIR). `https://www.apnic.net/about-apnic/organization/history-of-apnic/history-of-the-regional-internet-registries/`, December 2019.

[66] IP2Location. IP2Location LITE Databases. `http://lite.ip2location.com`, 2016.

[67] IP2Location. IP2Location databases. `http://www.ip2location.com`, 2017.

[68] IP2Location. IP2Location LITE Databases. `https://lite.ip2location.com/ip2location-lite`, 2019.

[69] Haiqing Jiang, Yaogong Wang, Kyunghan Lee, and Injong Rhee. Tackling bufferbloat in 3g/4g networks. In *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, pages 329–342, New York, NY, USA, 2012. ACM.

[70] Ethan Katz-Bassett, John P. John, Arvind Krishnamurthy, David Wetherall, Thomas Anderson, and Yatin Chawathe. Towards IP geolocation using delay and topology measurements. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, 2006.

[71] Ken Keys. Iffinder. `http://www.caida.org/tools/measurement/iffinder/`, 2018.

[72] Ken Keys, Young Hyun, Matthew Luckie, and kc claffy. Internet-Scale IPv4 Alias Resolution with MIDAR. *IEEE/ACM Transactions on Networking*, 21(2):383–399, April 2013.

[73] S. Laki, P. Mátray, P. Hága, I. Csabai, and G. Vattay. A model based approach for improving router geolocation. *Comput. Netw.*, 54(9):1490–1501, June 2010.

[74] Peter Langfelder, Bin Zhang, and Steve Horvath. Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for R. *Bioinformatics*, 24(2):719–720, 2008.

[75] Yeonhee Lee, Heasook Park, and Youngseok Lee. IP geolocation with a crowd-sourcing broadband performance tool. *SIGCOMM Comput. Commun. Rev.*, 46(1):12–20, January 2016.

[76] Youndo Lee and Neil Spring. Identifying and aggregating homogeneous IPv4 /24 blocks with Hobbit. In *Proceedings of the 2016 Internet Measurement Conference*, IMC '16, pages 151–165, New York, NY, USA, 2016. ACM.

[77] Youndo Lee and Neil Spring. Identifying and analyzing broadband Internet reverse DNS names. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '17, page 35–40, New York, NY, USA, 2017. Association for Computing Machinery.

[78] Matthew Luckie, Bradley Huffaker, and k claffy. Learning regexes to extract router names from hostnames. In *Proceedings of the Internet Measurement Conference*, IMC '19, page 337–350, New York, NY, USA, 2019. Association for Computing Machinery.

[79] MaxMind, Inc. MaxMind GeoIP2 country databases. `https://www.maxmind.com/en/geoip2-country-database`, 2016.

[80] MaxMind, Inc. MaxMind GeoIP2 databases. `https://www.maxmind.com/en/geoip2-databases`, 2016.

[81] MaxMind, Inc. Maxmind GeoLite databases. `http://dev.maxmind.com/geoip/legacy/geolite/`, 2016.

[82] MaxMind, Inc. MaxMind GeoIP data correction. `https://support.maxmind.com/geoip-data-correction-request/`, 2020.

[83] Giuseppe Mazziotti. Is geo-blocking a real cause for concern in Europe? *Retrieved from Cadmus, European University Institute Research Repository*, 2015.

[84] D. Moore, R. Periakaruppan, J. Donohoe, and k. claffy. Where in the world is netgeo.caida.org? In *International Networking Conference (INET) '00*, Yokohama, Japan, July 2000. The Internet Society.

[85] James A. Muir and Paul C. Van Oorschot. Internet geolocation: Evasion and counterevasion. *ACM Comput. Surv.*, 42(1):4:1–4:23, December 2009.

[86] Austin Murdock, Frank Li, Paul Bramsen, Zakir Durumeric, and Vern Paxson. Target generation for Internet-Wide IPv6 scanning. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, page 242–253, New York, NY, USA, 2017. Association for Computing Machinery.

[87] Ashkan Nikravesh, David R. Choffnes, Ethan Katz-Bassett, Z. Morley Mao, and Matt Welsh. Mobile network performance from user devices: A longitudinal, multidimensional analysis. In *Proceedings of the 15th International Conference on Passive and Active Measurement - Volume 8362*, PAM 2014, pages 12–22, Berlin, Heidelberg, 2014. Springer-Verlag.

[88] NRO. Extended allocation and assignment reports. `https://www.nro.net/about/rirs/statistics/`, December 2019.

[89] OpenSignal. State of mobile networks: Usa (january 2018). `https://opensignal.com/reports/2018/01/usa/state-of-the-mobile-network`, 2018.

[90] Ramakrishna Padmanabhan, Patrick Owen, Aaron Schulman, and Neil Spring. Timeouts: Beware surprisingly high delay. In *Proceedings of the 2015 Internet Measurement Conference*, IMC '15, pages 303–316, New York, NY, USA, 2015. ACM.

[91] Venkata N. Padmanabhan and Lakshminarayanan Subramanian. An investigation of geographic mapping techniques for Internet hosts. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, 2001.

[92] Paul Pearce, Roya Ensafi, Frank Li, Nick Feamster, and Vern Paxson. Augur: Internet-wide detection of connectivity disruptions. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA.*, pages 427–443, May 2017.

[93] Ram Periakaruppan and Evi Nemeth. GTrace - a graphical traceroute tool. In *Proceedings of the 13th USENIX conference on System administration*, Seattle, Washington, USA, November 1999.

[94] Zachary N. J. Peterson, Mark Gondree, and Robert Beverly. A position paper on data sovereignty: The importance of geolocating data in the cloud. In *Proceedings of the 3rd USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'11, USA, 2011. USENIX Association.

[95] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. IP geolocation databases: Unreliable? *SIGCOMM Comput. Commun. Rev.*, 41(2):53–56, April 2011.

[96] USC/LANDER Project. Internet addresses geolocation dataset. `https://ant.isi.edu/datasets/geolocation/`, 2015.

[97] USC/LANDER Project. Internet outage measurements. `https://ant.isi.edu/datasets/outage/`, September 2017.

[98] USC/LANDER Project. Internet outage measurements. IMPACT ID `USC-LANDER/internet_outage_adaptive_a29c-20170702`, September 2017.

[99] USC/LANDER Project. Internet outage measurements. IMPACT ID `USC-LANDER/internet_outage_adaptive_a34c-20181001`, October 2018.

[100] USC/LANDER Project. Internet outage measurements. IMPACT ID `USC-LANDER/internet_outage_adaptive_a35c-20190101`, January 2019.

[101] Lin Quan, John Heidemann, and Yuri Pradkin. Trinocular: Understanding Internet reliability through adaptive probing. In *Proceedings of the ACM SIGCOMM Conference*, pages 255–266, Hong Kong, China, August 2013. ACM.

[102] Rapid7 Labs. Reverse DNS (RDNS). `https://opendata.rapid7.com/sonar.rdns_v2/`, September 2017.

[103] Researchscape International and Evergage, INC. 2019 trends in personalization survey report. `https://www.evergage.com/resources/ebooks/trends-in-personalization-survey-report/`, April 2019.

[104] Philipp Richter, Mark Allman, Randy Bush, and Vern Paxson. A primer on IPv4 scarcity. *SIGCOMM Comput. Commun. Rev.*, 45(2):21–31, April 2015.

[105] RIPE NCC. RIPE NCC measurements. `https://atlas.ripe.net/measurements/`, May 2016.

[106] RIPE NCC. OpenIPmap - a collaborative approach to mapping Internet infrastructure. `https://labs.ripe.net/Members/jasper_den_hertog/openipmap-a-collaborative-approach-to-mapping-internet-infrastructure`, February 2020.

[107] RIPE NCC. RIPE IPmap, geolocating Internet infrastructure. `https://openipmap.ripe.net/`, February 2020.

[108] John P. Rula and Fabian E. Bustamante. Behind the curtain: Cellular DNS and content replica selection. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 59–72, NY, USA, 2014. ACM.

[109] John P. Rula, Fabián E. Bustamante, and Moritz Steiner. Cell spotting: Studying the role of cellular networks in the Internet. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, pages 191–204, NY, USA, 2017. ACM.

[110] Quirin Scheitle, Oliver Gasser, Patrick Sattler, and Georg Carle. HLOC: Hints-Based Geolocation Leveraging Multiple Measurement Frameworks. In *Network Traffic Measurement and Analysis Conference (TMA)*, Dublin, Ireland, June 2017.

[111] Sayandeep Sen, Jongwon Yoon, Joshua Hare, Justin Ormont, and Suman Banerjee. Can they hear me now?: A case for a client-assisted approach to monitoring wide-area wireless networks. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 99–116, New York, NY, USA, 2011. ACM.

[112] Anant Shah, Romain Fontugne, and Christos Papadopoulos. Towards characterizing international routing detours. In *Proceedings of the 12th Asian Internet Engineering Conference*, AINTEC '16, pages 17–24, New York, NY, USA, 2016. ACM.

[113] Yuval Shavitt and Noa Zilberman. A geolocation databases study. *IEEE Journal on Selected Areas in Communications*, 29(10):2044–2056, 2011.

[114] Satinder Pal Singh, Randolph Baden, Choon Lee, Bobby Bhattacharjee, Richard La, and Mark Shayman. IP geolocation in metropolitan areas. *SIGMETRICS Perform. Eval. Rev.*, 39(1):347–348, June 2011.

[115] S. S. Siwpersad, Bamba Gueye, and Steve Uhlig. Assessing the geographic resolution of exhaustive tabulation for geolocating Internet hosts. In *Proceedings of the 9th International Conference on Passive and Active Network Measurement*, PAM'08, pages 11–20, Berlin, Heidelberg, 2008. Springer-Verlag.

[116] Skyhook-Wireless. Skyhook location SDK overview. `https://www.skyhook.com/`, January 2019.

[117] Joel Sommers and Paul Barford. Cell vs. WiFi: On the performance of metro area mobile connections. In *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, pages 301–314, New York, NY, USA, 2012. ACM.

[118] Neil Spring, Ratul Mahajan, and Thomas Anderson. The causes of path inflation. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 113–124, New York, NY, USA, 2003. ACM.

[119] Neil Spring, Ratul Mahajan, David Wetherall, and Thomas Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Trans. Netw.*, 12(1):2–16, February 2004.

[120] Statcounter.com. Mobile and tablet Internet usage exceeds desktop for first time. Press release `http://gs.statcounter.com/press/mobile-and-tablet-internet-usage-exceeds-desktop-for-first-time-worldwide`, November 2016.

[121] Meenakshi Syamkumar, Ramakrishnan Durairajan, and Paul Barford. Bigfoot: A geo-based visualization methodology for detecting BGP threats. In *2016 IEEE Symposium on Visualization for Cyber Security, VizSec 2016, Baltimore, MD, USA, October 24, 2016*, pages 1–8, 2016.

[122] Team Cymru. IP to ASN mapping. `http://www.team-cymru.org/IP-ASN-mapping.html`, August 2016.

[123] The Internet Society. Special-use IPv4 addresses. *RFC 3330*, September 2002.

[124] Sipat Triukose, Sebastien Ardon, Anirban Mahanti, and Aaditeshwar Seth. Geolocating IP addresses in cellular data networks. In Nina Taft and Fabio Ricciato, editors, *Passive and Active Measurement*, pages 158–167, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[125] Alex Varshavsky, Eyal de Lara, Jeffrey Hightower, Anthony LaMarca, and Veljo Otsason. GSM indoor localization. *Pervasive Mob. Comput.*, 3(6):698–720, December 2007.

[126] Visualware. VisualRoute, traceroute and network diagnostic tool. `http://www.visualroute.com/`, June 2013.

[127] W3C. Geolocation api. https://www.w3.org/TR/geolocation-API/, April 2018.

[128] Yong Wang, Daniel Burgener, Marcel Flores, Aleksandar Kuzmanovic, and Cheng Huang. Towards street-level client-independent IP geolocation. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, pages 365–379, Berkeley, CA, USA, 2011. USENIX Association.

[129] Bernard Wong, Ivan Stoyanov, and Emin Gün Sirer. Octant: A comprehensive framework for the geolocalization of Internet hosts. In *The 4th USENIX conference on Networked systems design & implementation*, 2007.

[130] Yinglian Xie, Fang Yu, Kannan Achan, Eliot Gillum, Moises Goldszmidt, and Ted Wobber. How dynamic are IP addresses? In *Proceedings of the ACM SIGCOMM Conference*, pages 301–312, Kyoto, Japan, August 2007. ACM.

[131] Qiang Xu, Junxian Huang, Zhaoguang Wang, Feng Qian, Alexandre Gerber, and Zhuoqing Morley Mao. Cellular data network infrastructure characterization and implication on mobile content placement. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '11, page 317–328, New York, NY, USA, 2011. Association for Computing Machinery.

[132] Uri Yacobi-Keller, Evgeny Savin, Benjamin Fabian, and Tatiana Ermakova. Towards geographical analysis of the autonomous system network. *International Journal of Networking and Virtual Organizations (IJNVO)*, 21(3):1–18, July 2019.

[133] Kyriakos Zarifis, Tobias Flach, Srikanth Nori, David Choffnes, Ramesh Govindan, Ethan Katz-Bassett, Z. Morley Mao, and Matt Welsh. Diagnosing path inflation of mobile client traffic. In *Proceedings of the 15th International Conference on Passive and Active Measurement - Volume 8362*, PAM 2014, pages 23–33, Berlin, Heidelberg, 2014. Springer-Verlag.

[134] Ming Zhang, Yaoping Ruan, Vivek Pai, and Jennifer Rexford. How DNS misnaming distorts Internet topology mapping. In *Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference*, ATEC '06, Berkeley, CA, USA, 2006. USENIX Association.