Geolocation through peer data

Independent degree project - Master of Science in computer engineering

Pontus Wendle Ekholm

MID SWEDEN UNIVERSITY

In

Examiner: Tingting Zhang, tingting.zhang@miun.se

Supervisor: Patrik Österberg, patrik.osterberg@miun.se

Author: Pontus Wendle Ekholm, poek1000@student.miun.se

Degree program: M. Sc in engineering - computer engineering, 300 ECTS

Main field of study: Computer engineering

Semester, year: St, 2017

Abstract

Internet geolocation has so far only been accurate down to a couple kilometers, depending on the area of the IP address that is being located. This is often implemented as a database of hosts with locations to combat the fact that absolute delay measurements are unstable.
In this report a method which uses the topological information of a network has been proposed and tested on the region of Mid Sweden. Multiple users of a Internet based news service supplied their physical addresses and their IP-addresses. Tracerouting from a static location revealed that users in close, physical proximity of each other share many hosts in their traceroute. These traceroutes to the different users was then modeled as a graph and an algorithm that estimates the location of an unknown user by aggregating the location data of users sharing the same branch was later implemented and tested.
Testing was done by partitioning the data into a training and testing set. The results of this test were later evaluated allowing the new method to be compared to a current, database driven method in use by the news distributing company.
The results revealed that if a measurement of the standard distance deviation is used for the new method, a threshold value can be calculated where the new method performs better. When this threshold value was applied the maximum, mean, median and minimum error of the new method was smaller than the older, database driven one. However, the instances where the new method performs better was just a small fraction of the original data set.

Keywords: IP-geolocation, tracerouting, graph database

Table of Contents

1Introduction

The problem of IP (Internet Protocol) geolocation, locating a device's geographic position based on the IP-address's metadata, is described in this chapter. First a definition of what it is and what metadata can be used, then a historical review of geolocation methods and finally a rough description of this projects chosen method to determine a geographic position from IP-address metadata.

1.1Background and problem motivation

IP geolocation is defined as finding the physical position of a host based on its IP-address [1]. Examples of an IP-address's metadata to estimate a users position, without intrusion, is the IP-address class, the communication delay and the DNS name an interface has been assigned.
It's worth mentioning that a service could just directly ask a user to submit his or her location. With an Assisted GPS-equipped device the location would be known with an error of less than 60 meters even if the user was moving [2] but this is not always the case, and asking a user to work out their location and then submit it can lead to a slower user experience. The latter is also vulnerable to users submitting false locations, which can be mitigated somewhat by quality assurance.
Geolocation is not a new problem and is often implemented with a database of known locations [3] and assigning the user to one of them given which subnet she is a part of because it is easy to implement, it scales well and the granularity of the estimation can be known to the user much easier. But when sub-networks can have nodes hundreds of kilometers apart the error is large [4]. The accuracy is affected by the granularity of the entries and continuos updates to the database, which means that the user has to be careful when selecting a database driven service to make sure it meets the requirements on granularity, accuracy and performance all of which has to be tested with new data.
There are other geolocation methods beside the database driven that are measurement based instead. Some IP-addresses locations can be known, and by measuring the delay from these addresses to the target IP-address one can use a

triangulation algorithm to abuse the fact that the propagation delay is proportional to the distance between the sender and the receiver [5].
One can also enhance the performance by using landmarks when there are not enough vantage points close by. The edge vantage point can assign the IP-address the location of the closest landmark with comparable delay.
The problem of geolocation has had many breakthroughs and most of them are based around the insight that the more you know about the client's local area, the easier it is to narrow down its location [6]. The work already done has shown that merely measuring the time delay and multiplying it with the signal's speed has a high margin of error because networks have congestions, are often not fully meshed, often have only a finite number of nodes that rarely match the number of square kilometers in an area, and are also unreliable meaning that packets needs to be retransmitted and often takes different routes for different transmissions over the between the same hosts.
Taking the network topology into account reduces this margin of error considerably, but that still means tens of kilometers of error. That is enough to find out which county someone is located in, or municipality for the most sparsely populated places, but a breakthrough was made about five years that used relative delays. In some areas the margin of error could be reduced to under a kilometer, enough to pinpoint which area of a city someone was located in.
Mittmedia, a distributor of news through multiple Internet platforms, finances itself primarily by localized ads. The purchasers of these ads are often businesses active in just a few places in Mid Sweden, and since the ad space is limited, it is important that readers see ads that are relevant to them and the area in which they live. There are problems with localization when the user reads an article shared by someone on Facebook as neither the Facebook browser nor the normal browser offers GPS access under direct control of the service.
Some web content needs a subscription to one of Mittmedia's papers and this subscription needs a physical address. Mittmedia's own app also allows users to submit their address and other geographical areas the user might be interested in. This information can be used to personalize the ads, but a lot of content that Mittmedia provides can be read without a subscription and many of those who read the content do not supply their current position meaning that IP-based geolocation has to be used in order to determine which ads are relevant to the

users location (e.g, a store hundreds of kilometers away might not be relevant at all).

It's important that the storage of this highly personal data follows strict ethical guidelines and Mittmedia has gathered the data in preparation of this study under a signed, written user agreement from their users that the data they submit can be used for purposes of research dealing with their current technology and needs.

1.2High-level problem statement

Instead of using universities or companies as landmarks for this problem, users of a particular company with known locations will be used as landmarks since users can move and make adjustments in their connectivity and are not as static, meaning this is an opportunity to study if that makes a significant difference when estimating a position.. Also, since the area of interest is not the entire globe but a geographically constrained area, vantage points can be physically placed at strategic locations. This means that this particular implementation does not necessarily perform well outside the geographic constraints of the users but it could give insights for implementing a similar system in other geographic regions as long as one is mindful of the region's characteristics.
The question is if this method will work better than the database driven service already available to Mittmedia, for the region of Mid Sweden specifically. The precision obtained by the current method is tens of kilometers, but it is almost entirely thrown off by centralized Internet service distribution since a user can be positioned at a major router hundreds of kilometers away. By using the topological information, available by tracerouting the IP-addresses provided from the user data and mapping out the topology of the paths between vantage points and the users, a better system could be possible or at least one that's also viable.

1.3Scope

The focus of this project is on creating a system that finds the location of a client based on the trace route from a VP to a client. This algorithm will be tested on IP-addresses that have only one observed user in order to combat the dispersion that might be introduced by networks using Network Address Translation (NAT) to assign more than one host to a single IP-address, and thus

there exist no functional relationship between IP-address and physical address for fine granularities.

Optimizing the placement of vantage points and making them secure is outside the scope of this project, as is also assuring the quality of the data provided. All things known about the data is what address a user has and what IP-addresses are observed only with them.

All IP addresses were only associated with locations in the Mid Sweden region as defined by the Swedish National Board of Transportation [7]. This means that the results are not applicable to any given IP-address in Sweden, let alone the world.

Because of firewalls and other problems with direct ICMP communication the project will focus on topology based estimations in the event that routers are unresponsive to direct measurement of communication delays between hops.

1.4Concrete and verifiable goals

The end result will be the comparison between the current, popular system and the proposed algorithm. The comparison will focus on the errors of the algorithms and the distributions of the error. The goals are;

- Make a viable geolocation method using only user submitted and trace route data.
- Analyze the maximum, minimum and average error and compare it to a service currently in use.
- Analyze the distribution of the error and compare it to a service currently in use.

1.5Outline

Chapter 2 deals with database driven methods of geolocation in general and measurement based geolocation methods with Constraint Based Geolocation (CBG) in particular. It then follows up by describing a way to model traceroutes as mathematical graphs and gives a description of

Chapter 3 describes the data available and how it was acquired. It also describes how it the data was partitioned into training and testing sets, while also describing how the data was used for training and testing respectively. Finally, the chapter mentions how the test results were evaluated.

Chapter 4 describes the technical implementation of the vantage points, the test result evaluation script, the storage of the trace route data and vantage point deployment.
Chapter 5 contains a summary of the test results provided by the evaluation script and illustrations detailing the error distributions of the methods.
Chapter 6 contains conclusions about the geolocation method using trace routes when compared to the one in current use. It also contains an ethical discussion about user data storage and transparency when sensitive user data is analyzed and stored.

2Related work

In this chapter a measurement method is described and compared with database driven methods, which will provide insights about how to implement similar methods even if not all conditions for it are met. The chapter describes how a trace route can be modeled as a graph and how the graph properties can be used for geolocation purposes, such as providing the precision of an estimation.

2.1Database lookup of IP location

This method uses list of IP addresses and physical locations associated with them. It is often implemented as API where a request contains an IP address and the response can be country, county, city and zip code containing the location of the host because the location of an IP-address is assumed to be static and thus does not need continuous updates. The granularity of the entries often vary between regions and services meaning some regions can provide the zip code while others might only reveal the country. Another name is "registry based geolocation". [3]

This method has an advantage in that the locations are already supplied, meaning that the time to find an estimated location of the IP address is only dependent on the request to the database. A big disadvantage is that there is often no way to know how good this estimate is [3]. For example, if a city and street address is returned without a margin of error there is no way to know if the IP address could be a couple of blocks away in the worst case scenario or in an entirely different country hundreds of kilometers away. Another disadvantage is that the entries have to be updated since the location of the IP address relies on one entry and is not the result of several records being aggregated [4]. The database driven method assumes that updates doesn't need to happen continuously and that eventual consistency is good enough for most requirements.

The database driven methods achieve about 50% accuracy in determining the city of a host and 99% accuracy in determining the correct country, but the figures vary between geopolitical regions [4] [8].

Examples of these services are WhoIs [9] and IP2Location [10]. These services look up which entity has registered the IP-address, which is often an ISP

(Internet Service Provider) and can give a street address, city and country that has been assigned the IP-address block, but often this only provides the address of the ISP's data center. [Spotter paper] In many cases additional information supplied by the ISP offers more granularity, but it is often at the discretion of the ISP; how great the error can be is mostly unknown.

2.2 Constraint based geolocation

In order to estimate a hosts location one can use this method, which somewhat mirrors GPS. It is based around measuring delays from known points and finding an area that intersects with all the circles radiating from the points. The simplest implementation of this algorithm simply sets the hosts position to the centroid of the area intersecting all the circles radiating from the points.
In order to measure the distance from the any of these points some sort of action has to be taken by either the hosts or devices at the individual points to "discover" each other, giving them the points the name "vantage points" (VP) [5]. They either send a signal to a host that is supposed to be geolocated, or they receive a signal from the host and sends a response, and the delay is measured and recorded by the system. They can also be given the task to perform a traceroute, in which case they send a succession of ICMP (Internet Control Message Protocol) packets with increasing values of the parameter "time to live" which determines how many hops between hosts the packet is allowed to travel, and when the packet is not allowed to travel anymore the host it stopped at sends a packet back to the sender stating where the packet "died". This chain of hosts is called a "traceroute". Any computer that can send and receive ICMP packets with a stable Internet connection can serve as a VP. Traceroutes can also be implemented with other types of packets, such as TCP.
Another possible addition to the system is landmarks, denoted as “L”. A landmark is a functional relationship between an IP-address and a physical address. The granularity of the address can be everything from a continent to a city to a GPS-position. These landmarks can be used within a region that contains the host, but is still too large, to narrow down the position further by looking at commonalities between the target and the landmarks.
In theory, the speed of a packet on the Internet is 2/3 the speed of light, but because of queuing delays, the fact that the Internet is not a full mesh, that packets does not always take the same route makes between two hosts and other factors means that in practice it doesn't always work that well.[6]

The approximation of the distance is then used as a radius from the VP. Each VP is then the center of a circle and only geographic locations within the intersection of these circles are considered, which will serve as the constrain for the localization.

This method has a median error of tens of kilometers[5] which is useful for finding the country, or maybe administrative division, that contains the point. There are ways to improve the overall accuracy of this algorithm. One can use the topology of the targets subnet to estimate how big of a difference last-mile-delays are making. The number of hops is loosely affected by geographic distance. This is however not the case in more countries with better communications infrastructure. Topology aware geolocation techniques, however, are more susceptible to tampering by malicious attackers who wants to avoid detection [11].

One can also use population density maps, i.e setting the users location to the most densely populated place or the population center (the average longitude and latitude for every inhabitant) within the region that intersects with the CBG [12]. These methods, however, are best suited for enhancing already established, tested and reliable methods of geolocation since the utility of these maps is highly dependent on the error and/or granularity of the method. Population density maps could also vary in their accuracy between regions, and the desired resolution could be impossible to find in one region but not in another, neighboring region. The access to population density information, wether through maps or other formats, for commercial purposes is also dependent on the regional government, as is the granularity of said data and wether or not the information is machine readable or needs formatting to work as a service. For a service who's only purpose is to work in a geographically constrained region it could be a good addition.

One can also use the relative delay between landmarks and the target and simply set the location to the landmark with the lowest relative delay since relative delay has a better correlation with distance than absolute delay [6]. Relative delay is measured from a common router between a landmark and the target when a trace route is done on both. The estimated distance is then based on the delay from the common router to the landmark plus the delay from the common router to the target [6]. This method needs an unobstructed way to send packets between the vantage points and the target, as well as a way to perform traceroutes, or it is not possible to implement it meaning that firewalls

are a real threat. Depending on the VP density in the areas containing the target host, number of VPs available and the number of landmarks in physical proximity to the target host the median error could potentially be less than a kilometer [6].

2.3Model of traceroutes

The routes a packet travels from a fixed host (the root) to many different hosts can in theory be modeled as a tree, which have the same properties of a multicast tree [13].

The nodes in the tree are the hosts and each branch is a different route the packet takes. The users, the end of a route, are then defined as the leaf nodes of the tree. This model allows the implementation of a simple algorithm to initiate a start node and continue down the tree to gather all leaf nodes. These leaf nodes are then used to approximate the location of an IP-address. The start node is first set as the host where the trace route ended, node n. If no nodes are found, set the start node to the second last host, n - 1. Continue until the branch of the trace route yields at least one leaf node.

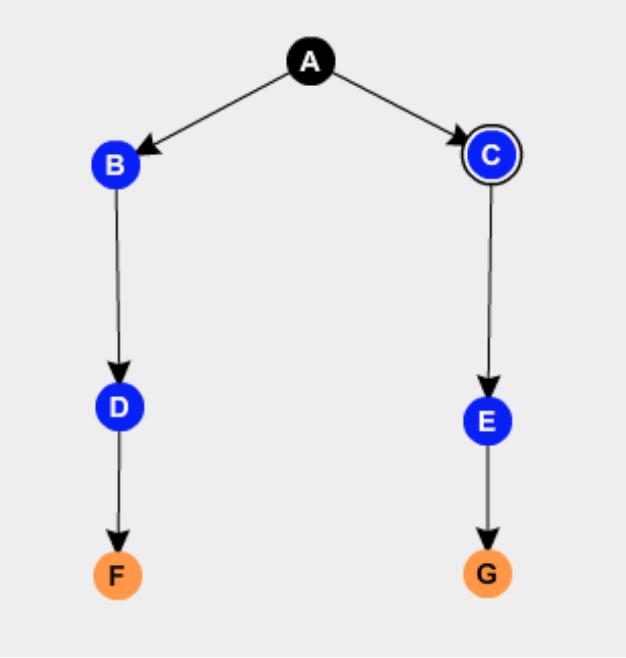

Figure 2.1: This graph meets the criteria of a tree. The root node, A, and the blue nodes have relationships. The algorithm would return the orange nodes, who have no relationships and can start at any node without problems.

As seen in figure 2.1, the algorithm would yield a set of distinct nodes and terminate properly if any of the nodes were chosen as the starting point for the algorithm. If the root node, A, was chosen it would traverse down to nodes B and C. In the next run, the the algorithm would see that B and C have children and not return them, but look at their children instead, D and E. They also have children, so the algorithm would continue down the tree and look at F and G. Since they do not have any children, the algorithm would return F and G.
This model of routes is complicated by the fact that packets don't always travel the same route to it's destination, and a trace route is a succession of sent packets. There is a significant probability of cycles being formed because of detours, firewalls and dynamic IP-addresses.
The graph in figure 2.2 presents a big problem because traversing a path until one yields a node without relationships, starting at node B for example, would mean that the algorithm would traverse to node D, then to node C, then node E, then back again to B. Unless the algorithm has detected that it has returned to the starting place, it will never halt.
To make the algorithm still applicable, we can serialize the nodes in order to detect cycles and make sure that we neglect traverse already visited nodes.

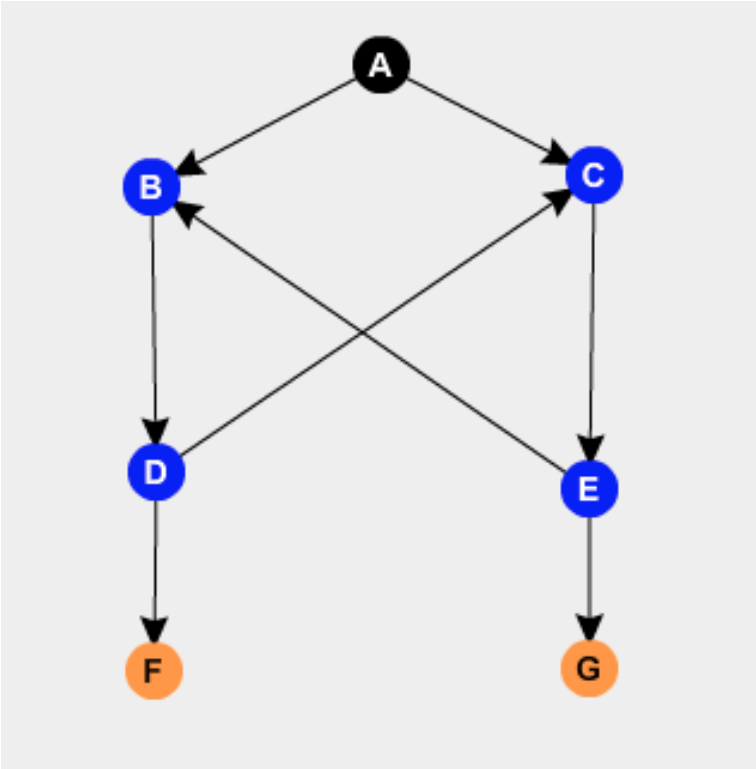

Figure 2.2: This graph is not a tree and the algorithm would enter an infinite loop.

2.4Graph databases

A graph database is a type of NoSQL databases that stores nodes and their relationships [14]. They were developed to store data that model a system that itself is often modeled as a graph, such as social networks.

A graph database links its nodes directly with each other through serializable relationships which is in contrast to a relational database that need three tables to store a many-to-many relationship between two tables, and JOIN operations in order to query data spanning multiple tables.

Instead of using a third entity to link two records, eg. students and courses, a student node would have relationships to the individual courses and querying each course would be made by first selecting the student and then select the nodes the student is related to through "enrolled". In the same example, in order to query all students enrolled in a course one would select all relationships of the type "enrolled" referencing the course and then select the nodes owning those relationships.

There are different graph databases released such as Titan DB, Neo4j and GraphDB [14]. Titan DB can be implemented on Apache Cassandra and have BerkeleyDB backend and supports a query language called Gremlin. With BerkeleyDB the database transactions support Atomicity, Consistency, Isolation and Durability (ACID) but with Cassandra only eventual consistency is supported. However, on a Cassandra back-end it's possible to scale without a single point of failure [15].

GraphDB combines it's database with a semantic analysis tool to use the data for machine learning in less complicated, specifically text analysis. It has support for ACID from the start and has support for the query language SPARQL (a recursive acronym of SPARQL Protocol And RDF Query Language). [16]

Neo4j is an open source graph database with ACID support out of the box and support the query languages Gremlin, Cypher and SPARQL [14]. However, in contrast to TitanDB it does not offer scalability on some other platform and possesses the problem of single-point-of-failure [14]. The Cypher query language is also not as general as Gremlin and works best with less complicated queries since the language is more focused on being expressive, borrowing inspiration from ASCII-art. It does however come bundled with a web based interface for database administrators.

All graph databases can be considered a form of NoSQL database [17]. Other types of NoSQL databases, such as document stores, are also considered NoSQL which also don't need three tables for many-top-many relationships between two entities and can also operate without a centralized schema catalog. The most popular is MongoDB [18], used with the node.js back-end, which stores data as JSON-objects in JSON-documents.

MongoDB is not entirely schema less since every document needs a schema but it does not need a centralized schema catalog like relational databases do. In order to express many-to-many relationships a JSON-object can store the other objects it is related to as member variables, or as objects in a container. Representations can also be nested, as in records containing records. However, in the case of MongoDB, this means that transactions among multiple documents is not possible and therefore there is no direct support for ACID transactions, only eventual consistency achieved through replicating documents and either accepting out of date documents or fetching from the closest store (determined by ping delay) in order to have access to more current data. [19]

One other downsides with MongoDB that graph databases don't have is that when implementing the schemas for the documents the database admin needs to have prior knowledge about what queries it has to support since transactions across multiple documents are not possible and therefore records that are intimately associated with each other needs to be in the same document since they can't be joined. This means that the database admin cannot be application agnostic but must design the schemas according to the user requirements, a restriction the relational database and the graph database don't have since they can support ACID transactions. [20]

There are cases where graph databases make more sense than relational databases, such as models of social networks and the reason seems to be that the queries are graph problems where queries are of the nature "how many hops are there between these two nodes" and "which nodes can be reached from this other node within x hops" [14]. The fact that relationships are serializable and directly link discrete nodes makes it simple to find a node's relatives and also much less intensive to store. Because the relational database handles relationships between tables with other tables the storage of new entries increases much faster for each entry than with graph databases [14].

2.5Standard distance deviation of the 2D shapes

The users are mapped on a shape bounded by the data points, with the latitude as y-coordinate and longitude as x-coordinate. However, the Earth is not flat so in order to calculate the distance between two points accurately you must use the Haversine formula [21]. Since the Earth's radius varies one must use an estimate, meaning that it's accurate to about 0.5%.

Since the user's position is estimated from a number of points that form a shape, the size of the shape gives a clue of how big the standard error is.

This method can not only be used to find the distance between two points, such as the estimated location and the actual location, which could yield an error for example but can also be used in an aggregated method to compute the scattering of locations around a certain hop. The standard distance deviation, *sd*, is computed by taking the mean of the distances between the centroid, *c*, of the locations around a hop and the individual points, $z_1, z_2, z_3, \ldots, z_n$ and calculating the mean of the distances in the form $sd = \frac{1}{n}\sum_{i=1}^{n} \|z_i - c\|$ This should give a rough estimate of how good a hop is for estimating the location of a user. A smaller the standard distance deviation should be correlated with a higher user density.

3Methodology

This chapter describes how the user data was provided and partitioned into a training and testing set and how these sets were used for their respective purposes. It finally mentions how the test results were evaluated.
The choice to partition the users instead of the individual IP-addresses into a training and testing set is because the goal of the system is to find unknown users who have never submitted a location to be found.

3.1Data gathering

The user data will be provided by the media distributor, Mittmedia. It will consist of an event log which will log the users IP-address and user id along with a table of users physical addresses. The current service for geolocation will be accessed, and IP-addresses associated with exactly one user id assigned a location by the geolocation service as an estimated location and the users submitted address and set as the users true location.
The user IDs will then be divided randomly into two sets; one for training and one for testing. All IP-addresses associated with the users who were sampled for training will then be placed in a training set and the rest of the IP-addresses were stored in a testing set.
In total 51 users will be used for training and 29 users for testing, which adds up to 80 users in total. How the users are divided into training and tested will be described in chapter 4, implementation, but it involves stochastic processes which is the reason why the testing and data set size proportions will only approximate the rule of thumb, 67% training and 33% testing, outlined in the implementation chapter.
The reason so few users will be used is because only IP-addresses associated with one user were allowed to serve as landmarks to retain a functional relationship between IP-address and physical address.

3.2Training with trace routes

The IP-addresses for training will be trace routed and each hop saved as a node in a graph data base to make traversing a particular branch easier for testing and estimation purposes. Each hop will have a number of unidirectional relationship

with hops immediately succeeding it in trace routes and also have a unidirectional relationships with users if the hop was the last hop of a trace route.
The nodes modeling the hops will only contain their IP-address as a property while the users have their physical address's longitude and latitude along with their user id as a property.

3.3Testing with trace routes

After training, the IP-addresses in the test dataset will successively be trace routed. When a trace route ends all users associated with the last hop will then be fetched from the database and used to determine a location. If no users are matched, the second last hop of the trace route is used to match users sharing it and if that also results in no users the algorithm continues trying to match users to the hops until finally terminating at the first address of the trace route.
The users sharing the hop are then used to estimate a location for this new user.

3.4Evaluation

Evaluating the algorithm requires the algorithm's estimation of a users position to be grouped together with the previous service's estimation and the user submitted address.
To calculate the algorithm's performance the distances between the predicted location and the submitted location will be used as the error for the respective methods. The distances will be calculated in a way that conforms to the nature of geography, such as the great circle distance given by the Haversine formula. In every case the value of the standard distance deviation of the new method will be calculated and recorded, so as to give a clue of how good the new method's estimate is.
The test results will then be evaluated and a threshold of the standard distance deviation will then be calculated to filter out estimates from the new method that are not reliable enough, since they have a large mean of the errors. This filter will be calculated iteratively by decreasing the tolerance of the standard distance deviation of the new methods predictions by set incremental steps while monitoring the descriptive values of the respective methods' performance i.e minimum error, mean error, median error maximum error and percentage of times one method gave an estimated location that was closer to the user

submitted location. This will reveal if the standard distance deviation is a useful metric to construct a filter that improves the new method. If there is a threshold that results in better performance for the new method, it will be used to filter rows based on unacceptable estimated locations by the new method and the distributions of the two cases (with and without the filter applied) will be analyzed and conclusions drawn later.

This evaluation will reveal keys about the performance of the two methods for this geographic region. It will not reveal the performance in the general case but if this region is the most significant one it should offer guidance when choosing a geolocation method.

The test results and the evaluations will later be compiled into a report.

4Implementation

This chapter describes the technical implementations of the project parts. It first describes how a vantage point was implemented. It follows it up with describing how a trace route was stored, in this case it mentions some technical aspects of the graph data base. It then mentions the algorithm for estimating an unknown user position by traversing the graph and using related nodes. It finally mentions how the script that evaluates the test results was implemented.

4.1Vantage Points

The Vantage Point were implemented on microcomputers capable of running simple Python scripts. It was each given a public IP-address to be reachable directly for ease of configuration, updating and to start tests. It was not possible to give the VP a static IP-addresses, instead broadcasting its IP-address every ten minutes to make sure that consistency was kept. If no broadcasts were received it was considered dead.

A trace route software was implemented in Python with the routes stored in a graph database.

When assigning a geolocation to an unknown host, a modified script could instead traverse the nodes to find related users.

4.2Dividing users into training and testing sets

It has been known since the 30s that training and evaluating an algorithm with the same data leads to overoptimistic results [22]. Testing the algorithm on new data, however, is a much better predictor of the typical performance of the algorithm once it is in use since it avoid combats overfitting [22].

In practice, new data is often not available but this problem can be mitigated by splitting the data set into two data sets: one used for training and the other one used for testing.

Before tracerouting a script was written to split the users into two sets, one for training and one for testing. The user IDs are stored in a list and iterated over, each having a 67% probability of being selected for training. Because of the small number of users one set is not necessarily twice the size of the other.

This cross-validation procedure is also known as the "hold-out" method and is the simplest to implement since it involves only a single split of the dataset [22]. It is not without flaws though as the variance of the errors across multiple repeats with data randomly split among the two sets is large. The results of the test depends on what data ended up in which data set.

Two other popular cross-validation procedures are k-fold cross validation and leave one out. In k-fold cross-validation (also known as "leave p out") the data set is randomly divided into k data sets and one of these is used for validation while the other k - 1 sets are used for training. This is repeated k times such that every subset has been used for validation in one of the test runs. The variance between runs is inversely proportional to the value of k and is therefore proven to have a lower variance than hold out [22].

In "leave one out" k is set to n where n is the total number of data points [22], meaning that a single data point is used for validation once. It has the lowest variance of the three methods mentioned since k cannot exceed n.

The hold out procedure is best chosen if there is only one training and test run possible since it requires only a single split and thus a single training and test run. The k-fold cross-validation procedure requires the algorithm to be run k times and is therefore k times slower, with leave one out being n times slower than hold out.

The reason the hold out method was chosen instead of k-fold cross validation or leave one out, despite the fact that the data sets are so small, is because of the training script's and the testing script's run time. In spite of the fact that there was less than a hundred users, each had multiple IP-addresses associated with it and therefore the number of individual data points for training was in the order of thousands. Since it took seconds to perform a traceroute it took hours to split the data set, train the algorithm and test it just once.

4.3Training and testing scripts

The traceroutes were performed according to the basic traceroute algorithm [23] were an initial ICMP packet was sent out with a Time To Live (TTL) value of 1 and then incrementing the TTL-value by 1 for successive packets until either the target host has been reached, there has been three successive time-outs or the TTL value has reached 30.

There are more modern approaches to the traceroute algorithm known since at least the early 90s [23] but the basic algorithm was chosen because of how trivial it is to set up.

The second condition of the traceroute, to break the traceroute after three successive time-outs is because a time-out could be the sign of a firewall. Three was set as the value because it is a standard in the Linux implementation of traceroute. This does not mean that three successive time-outs is a firewall, it could also be that the router does not respond to ping, but it is treated as a complete traceroute [24].

The time-out was set to 1 second which is smaller than the standard 5 seconds [24] but the time interval was chosen because of time constraints. Each training session performed a traceroute for more than 3 000 IP-addresses and the testing session performed traceroutes for half of that, but still in the range of 1 500 - 2 000 IP-addresses.

With a time-out of one second, a traceroute has a maximum time of 30 seconds (since the traceroute was aborted when the TTL-value reached 30) which means that with 3 000 IP-addresses the training could take a maximum of 33 hours and 20 minutes. Observations prior to full scale testing reveal that traceroutes are typically within the order of 10 hops long and the delay of each router's response was typically less than 20 milliseconds, meaning that the time to perform the training was typically within 3 hours.

The two sets were divided into two different files for ease of debugging and to assure that no test data was present in the training set and vice verse. Both the training and testing script iterated over each IP-address in each respective set and performed a traceroute. If the case of a lost Internet connection the scripts would terminate. Because of the implementation of a graph database, where duplicate entries can be handled by only allowing unique rows and unique relationships, the training script could be run again in an idempotent manner such that a traceroute already recorded would not be stored again, however a backup script was implemented that allowed a user to manually select a starting row for time constrain reasons. If the testing script failed and had to run again it would completely overwrite the old file with test results and start again because the test set was smaller than the training set.

4.4Trace route storage

The trace routes were stored in a graph database with hosts as nodes and routes as relationships in order to represent the model of a traceroute detailed in chapter 2.3. The graph database chosen was Neo4j because of the fact that it was open source, had ACID support out of the box, was accessible through both a REST-API and Python API, py2neo. The problems with scaling were not deemed a problem since the project was never meant to be moved into a production environment but rather test the geolocation algorithm through topological ways.

The queries were written in the Cypher query language because of the fact that only two complicated queries were written, most transactions were supported by the Python API

If two users had the exact same trace route, the route was stored exactly once with one extra relationship added at the end meaning that duplicate entries of hosts could be avoided. Duplicate data would occupy more space and if the route was ever changed, and stored more times than one, every instance would need to be updated to keep consistency. Two routes can also differ with few hosts, such as one hop out of ten, meaning that 90% of the data describing the second route would be redundant information.

Performance wise the graph database does not need to compile tables, join them, aggregate numbers and then sort them to get the desired output but can instead traverse the graph by using the relationships. This meant that one could get better performance and use less code than with a relational database since structured queries were the norm [25].

The occurrence of cycles was also a big problem for relational databases since the exact same a traceroute can be described in an infinite amount of ways through "hop-by-hop" notations but topologically only one way.

4.5Estimating position

Most routers did not respond to ping requests, meaning a key part of the CBG algorithm (measuring delays) could not be performed. This meant that there was no direct way to discern which users sharing the hop were the best suited to estimate the unknown users position, which is a critical step in the improved CBG algorithm. Because of the fact that topology can reveal facts about an IP-address's physical location another algorithm based entirely on topology was implemented to counter that fact. It requires that other users were queried based

on their traceroutes, with any given host along the traceroutes being the key to find relevant users to estimate the position.

Everyone sharing the hop had to be used for the estimate and therefore, their positions had to be aggregated in some way. The median of their latitudes and longitudes respectively were used instead of the mean in order to avoid a small number of outliers to pull the centroid too much from the more dense areas.

In order to measure how great the dispersion was between the users sharing the hop a value of the dispersion was calculated by first computing a centroid as the mean latitude and mean longitude, then taking the mean of the users distances from the center. The exact formula for the standard distance deviation, *sd*, is given in chapter 2.4.

4.6Handling cycles

Because the network is unreliable and packets can take different routes between the exact same pair of hosts depending on factors such as traffic, priorities, firewalls and reconfigurations cycles could form within the graph leading the model of a tree to be flawed. The algorithm could be stuck in an infinite loop if a cycle is not detected and broken out of.

In order to discover cycles, the nodes need to be serialized with unique identifiers. The starting node of this sub algorithm (i.e the last host of the traceroute) is set and a node's neighbors are defined as the nodes it is related to (unidirectionally, but this does not necessarily exclude a pair of nodes to be related with each other). The goal of the sub algorithm is to return all the user nodes within the subgraph originating from the starting node.

As the sub algorithm traverses through the graph (ideally in one direction, since the nodes only have unidirectional relationships, but there are no guarantees) towards the edge nodes it stores all the visited nodes. For each layer of the subgraph, the nodes of that layer returns their neighbors that have not already been visited. This is so that the graph ideally does not traverse backwards, since a pair of nodes in a relationship with each other could freeze the algorithm otherwise.

When all nodes are visited, there will be no layer nodes that are unvisited and all the visited nodes are then returned. These nodes can later be used for fetching the users associated with them by iterating over them and fetching their users.

The algorithm is described in thorough detail as pseudocode below:

```
get_subgraph_nodes(root):
    visited_nodes = [root]
    layer_nodes = visited_nodes

    while true:
        layer_nodes = neighbors_of(layer_nodes) - visited_nodes
        if len(layer_nodes) < 1:
            break
        visited_nodes = visited_nodes + layer_nodes
    return visited_nodes

neighbors_of(nodes):
    neighbors = []
    for n in nodes:
        for m in n.get_neighbors():
            neighbors.append(m)
```

As can be seen in the pseudocode, the algorithm is made up of three nested loops; one in the primary function, get_subgraph_nodes, that terminates if there are no neighbors of the nodes in the layer that haven't been visited, calling the function neighbors_of that iterates over all unvisited neighbors of the layer nodes (that becomes the new layer nodes) which in turn has with a third loop that iterates over all neighbors of a particular node to append them to the list that's returned. The algorithm thus has a relatively high complexity of $O(n^3)$ meaning that it works best in graphs where cycles are rare and localized.

4.7Deployment of VP

One of the planned VPs were deployed, in the city of Sundsvall. This meant that there was only one base of reference for the trace routes. There were plans to deploy VPs to the cities of Gävle, Örnsköldsvik and Åre but they were never deployed.

The deployed VP performed one traceroute for each IP-address because of time constraints. Because the network layer is highly unreliable with detours, firewalls, congestions and outright failures of routers two traceroutes between the same pair of hosts could differ significantly.

The VP did not perform any of the calculations, the data after the trace routes had been performed were later extracted and analyzed.

4.8Evaluation script

In order to evaluate the algorithm's performance a script was written that read and compared the predictions to the user submitted location. It computed and displayed the minimum error, median error, maximum error and the mean error in kilometers of the two methods and also the amount of times the new method was better than the old.

It also showed the distribution of the respective prediction methods in a histogram with a staple size of 5 in order to visually approximate the skew of the respective distributions.

A filter was calculated by disallowing measurements where the standard distance deviation was above a certain threshold. This threshold was lowered iteratively until the new method was performing strictly better than the old.

4.9Filter calculation

The algorithm provided each estimated physical position of an IP-address with a measurement of the standard distance deviation as a possible basis for a filter to accept only estimates within an acceptable standard distance deviation threshold. The filter's purpose was to examine what difference the standard distance deviation made, if any, on the performance of the new method since a larger than average mean distance from the centroid is an indication that the subregion associated with that particular last hop is larger than average and could be less suitable depending on the error acceptance

The filter was found through experimental means by incrementally lowering the threshold for the standard distance deviation and excluding those estimates with a standard distance deviation value above the threshold. Both estimates are excluded, the old and the new method, to study if either or both methods change with the standard distance deviation since it might hold key information about the errors of both algorithms.

The incremental step of five kilometers was chosen for each iteration and the results of the evaluation script studied to determine wether or not any of the methods could be said to perform better. Smaller steps, such as 1 kilometer, was discussed but not implemented after observations confirmed that most increments of five did not produce any significant differences between evaluations (in some cases, lowering the threshold by five kilometers resulted in no estimations being filtered out).

The two values monitored most closely, who served as the basis for when one method was performing better than the other, was percentage of times one method had a smaller error than the other (which was called "percentage wins") and the median error. The median was selected over the mean to make the evaluation more robust to outliers and to account for any skew the error distributions might exhibit (see chapter 5 for the histograms of the two method's errors and their estimated distributions).

When both of the "percentage wins" and the median were better for the new method than the old the threshold was said to be found and served as the filter's threshold. A final evaluation was then made where the filtered, new method was compared to both the filtered and the unfiltered old method. This was because the old method could be affected since the entire row containing the IP-address, the new method's estimated position of this IP-address and the old method's estimated position of the IP-address, is filtered out if the standard distance deviation value of the row is above the threshold.

5Results

This chapter summarizes the test results for the two main scenarios; with all estimations included and one where a threshold precision was placed to filter out poor estimations. The test results are provided with a results from the current method. The results are a summary of the minimum, mean, median and maximum values for the methods and charts illustrating their error distributions.

5.1Distribution of the users' positions

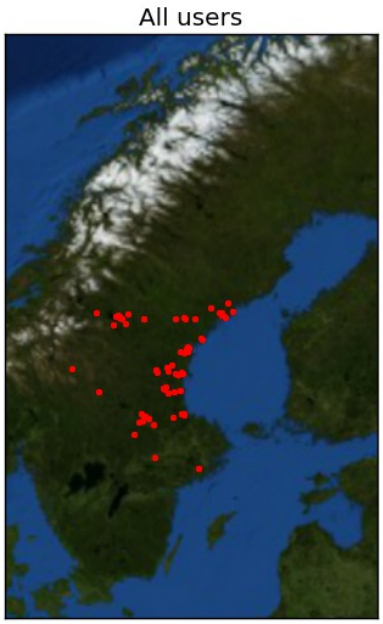

Figure 5.1: The positions of the users in the study.

As seen in figure 5.1 the users were largely constrained to the Mid Sweden region. The majority of the users were between Gävle and Örnsköldsvik but some were located close to Stockholm (which is outside the Mid Sweden region). Most of the users were located close to the coastal regions of the region but there is a non-significant number of users further inland, most notably the Östersund area.

The distribution is not uniform across the area but is somewhat clustered around the major urban regions of the official Mid Sweden region. However, since there are a lot of points outside the clusters there is a risk of variance caused by

outliers ending up in only one of the subset considering the complete dataset size of 80 users.

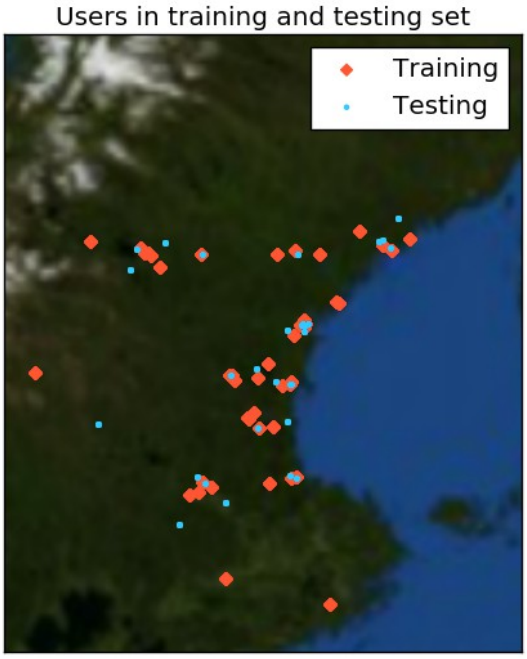

Figure 5.2: Distribution of users in training and testing set.

The random division of users into the training and testing set is illustrated in figure 5.2. Both the training and the testing set have users from the major urban areas of the Mid Sweden area represented in them as well as users from rural areas. However, the users in the training set span an area of Sweden that exceeds the users of the training set having users located in the Stockholm and Mälardalen area, as well as further inland into the Scandinavian Fjäll region close to the border of Norway. The testing set also has points tens of kilometers away from the closest data point in the training set but the area they span is completely enclosed in the area spanned by the data points in the training set. Both the training and the testing set has localized clusters centered around the urban areas, just like the complete dataset.

5.2Distribution of the estimated locations

The locations of the test users were confined to 8 points were all located in the eastern part of the Mid Sweden region around the two urban areas of Sundsvall and Hudiksvall, with a third cluster to the west of Hudiksvall. The urban areas of Gävle, Östersund and Örnsköldsvik were not represented among the

estimation points used for assigning the test users a location despite being the clusters having a clear presence in the training set. The estimation points' geographic position is illustrated in figure 5.3 with their standard distance deviations, relative to each other, being represented by the size of the circles. The size on the map of the standard distance deviation is not to scale but rather shows their position and their difference in size.

Figure 5.3: Estimation points used to assign a position for a users in the test set.

Table 5.1: Estimation points used for assigning locations to test users.

Latitude	Longitude	Standard distance deviation	Test users assigned to position
61.716869	15.589872	122.4 km	91
61.801330	16.972901	95.0 km	61
61.816627	16.526836	108.8 km	70
61.831924	16.561672	106.8 km	69
61.831924	16.826832	103.6 km	209
61.856351	16.759217	112.5 km	1323
61.958891	16.723012	0.0 km	70
62.404205	17.365733	0.0 km	137

As can be seen in table 5.1 most estimation points have a standard distance deviation of over 100 kilometers (5 out of 8). The point with the largest standard distance deviation, of 122.4 kilometers, is located outside any urban area with the closest village being Ljusdal. The estimation point with the second highest standard distance deviation, of 112.5 kilometers, is also outside any urban area and is actually located in a lake called Norra Dellen outside of Hudiksvall. This point was also the one most frequently assigned to test users, 1323 times out of 2030.

The two points with the lowest standard distance deviation, both at 0.0 kilometers, were located in two completely different areas. One was located in the community of Strömbacka, just north of lake Norra Dellen, and the other one was located in the city of Sundsvall which is the largest city in the Mid Sweden region.

5.3Results after training and testing

The results after the training trace routes were stored in a graph database and the test results were recorded in CSV-files with their two predicted locations (one from the old service and the other from the new algorithm) and the user submitted location. These results could later be evaluated and analyzed. Without the filter all 2030 testing instances done on 29 users were performed. When the filter was applied, only 268 instances had a standard distance deviation within the acceptable limits that covered 5 users in total.

5.3.1Test results with no filter

As seen in table 1, the old method has a larger maximum value of the error and a larger mean but has a better median value. The minimum error is also lower than the new method's and in more than half the cases the old method's location estimation is closer to the submitted location than the new method's.

Table 5.2: Method error measurements

	Old method	New method
Minimum error	0 km	2 km
Median error	32 km	131 km
Mean error	143 km	113 km
Maximum error	626 km	269 km

Percentage of times method performs better	62%	38%

In figure 1, both methods' distributions are presented. The new method's distribution is more uniform with no apparent skew while the older method has a noticeable skew to the left of the distribution, meaning that results within 5 kilometers are more common than results exceeding hundreds of kilometers.

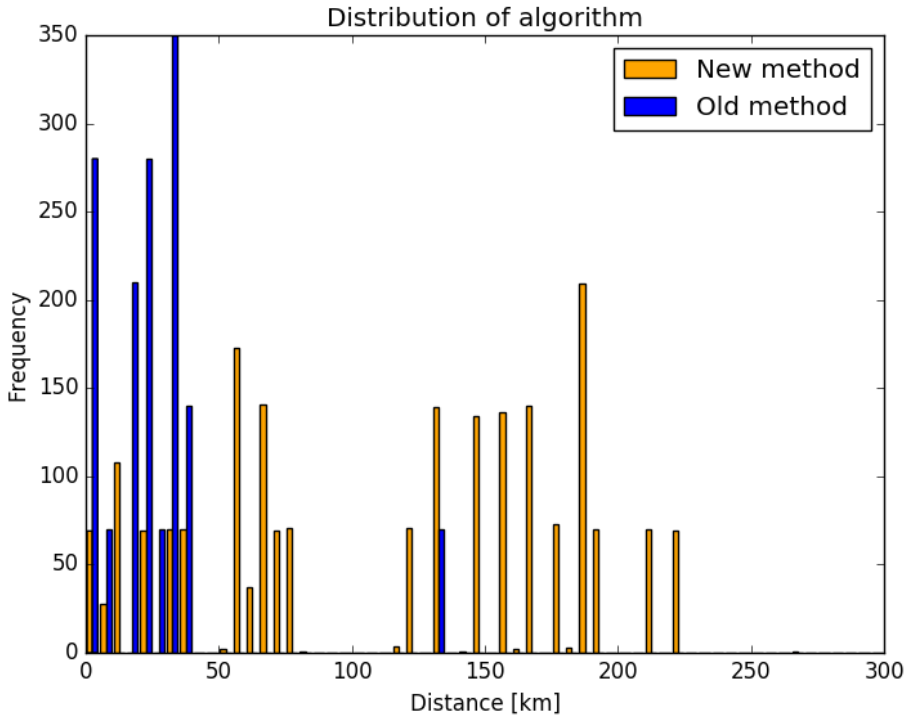

Figure 5.1: Distribution of method errors

5.3.2Test results with filters

The different tables illustrate different setting of the filter, i.e the maximum allowed standard distance deviation of a particular estimate of the new method.

Table 5.3: Method error measurements after filtering deviation above 120 km

	Old method	New method
Minimum error	0 km	2 km
Median error	32 km	131 km
Mean error	147 km	112 km
Maximum error	626 km	221 km
Percentage of times method performs better	61%	39%

As seen in table 5.3, filtering out the geolocations based on estimations with a standard distance deviation above 120 kilometers increased the performance of the new method relative to the old method by a percentage point, but the overall performance of both methods is largely unchanged. The maximum error has been filtered away from the new method's results but the mean is still unchanged by this.
Out of the 29 users, 1 test user's assigned location was based on an estimate with more than 120 kilometers in standard distance deviation. This means that with the filter in place, 1 of the users would not have an estimation that is beneath the threshold and would have to be handled specifically.
When the threshold was lowered to 115 kilometers it showed no changes since no estimations were filtered away (since the next largest standard distance deviation is 112.4 kilometers).

Table 5.4: Method error measurements after filtering deviation above 110 km

	Old method	New method
Minimum error	0 km	2 km
Median error	31 km	56 km
Mean error	77 km	76 km
Maximum error	425 km	189 km
Percentage of times method performs better	62%	38%

When applying a threshold of 110 kilometers the percentage of individual times one algorithm gives a better estimate than the other remains unchanged, however the overall performances of the algorithms have changed, especially for the new method. The median error is about half of what it was before applying this new filter, of the same magnitude as the old method, and the maximum error has also decreased to 189 kilometers from 221 kilometers.
However, out of 29 users 19 were not assigned a location based on an estimate with a standard distance deviation below the threshold.

Table 5.5: Method error measurement after filtering deviations above 105 km

	Old method	New method
Minimum error	0 km	2 km
Median error	31 km	38 km
Mean error	94 km	60 km
Maximum error	425 km	157 km
Percentage of times method performs better	51%	49%

As seen in table 5.5 the performance of the two algorithm's relative to each other has changed; one method outperforms the other in roughly half of the cases with the old method edging out the new method by 1 percentage point.
The old method has an increase in it's mean error while the new method has a decrease. The median error of the new method decreases while the median error of the old method remains unchanged. The maximum error of the old method remains unchanged while the new method's maximum error has decreased after lowering the threshold.
Out of 29 users 21 were not assigned a location based on an estimate with a standard distance deviation below the threshold.

Table 5.5: Method error measurements after filtering deviation above 100 km

	Old method	New method
Minimum error	5 km	2 km
Median error	34 km	32 km
Mean error	127 km	56 km
Max error	425 km	157 km
Percentage of times method performs better	38%	62%

The performance of the new method relative to the old method has improved as seen in table 5.5 as the new method gives a better estimated location of the test users than the old method in more than half of the cases. The median error of the new method with a threshold of 100 kilometer is the same as the median error of the old method unfiltered. The maximum error of the new method is however unchanged, as is the minimum error. The minimum error of the unfiltered old method remains smaller. The mean error of the new method decreased compared to the threshold of 105 kilometers.
Out of 29 users 24 were not assigned a location based on an estimate with a standard distance deviation lower than the threshold value.
In tables 5.3 and 5.5 one can see a improvement in the new algorithm relative to the old. All three tables reveal an overall performance increase for the new method with respect to the mean and maximum error while 3 and 4 reveals a decrease in the median error for the new method. The old method's mean error fluctuates while its median error remains relatively stable. The reason for the fluctuations of the old method's values is because of rows being omitted by the filter, since both method's estimated position are omitted when the row containing a standard distance deviation above the filtered value of the new method. The maximum error of the old method decreases in table 3 but remains at 425 km in table 4.

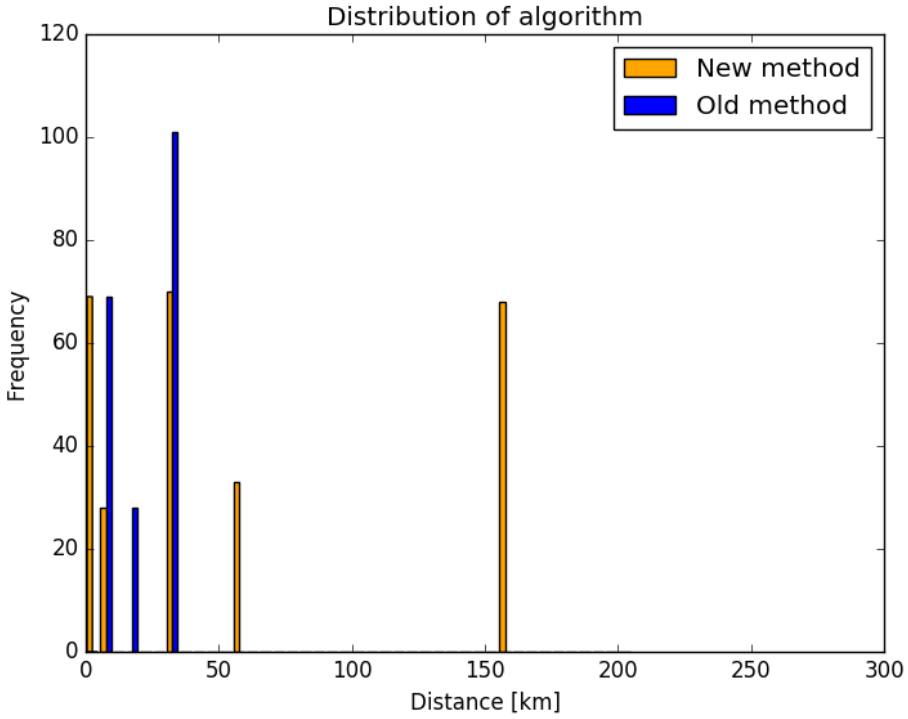

Figure 5.2: Distribution of old and new method after filter

Figure 5.2 illustrates that the new method has more estimations that are within 0 to 5 kilometers of their target than the old method, but the probability distribution of the old method is more skewed towards the left. There are barely visible values occurring across the histograms outside the most frequent values, which is why for aesthetic reasons the histograms does not extend enough to display the maximum values. The same is true for figure 1, where some values are significantly less frequent than others and the largest ones fitting this criteria have been left out of the histogram altogether.

5.4Distribution of user locations by difficulty of estimation

Some user positions were more prone to high errors than others. As seen in table 5.2 the maximum error of the old method is 626 kilometers while the new method has a maximum error of 269 kilometers. Most of the estimations from both methods have much smaller minimum values, though, at 0 kilometers of the old method and 2 kilometers of the new method with some errors appearing more frequently than others.

In figure 5.3 the user locations in the study have been plotted on the map with a color and shape given depending on if the attempts to locate a user with the new method at that position results in errors above or equal to and below the median

error of the method. This reveals if the errors of the attempted geolocation are dependent on the users location.

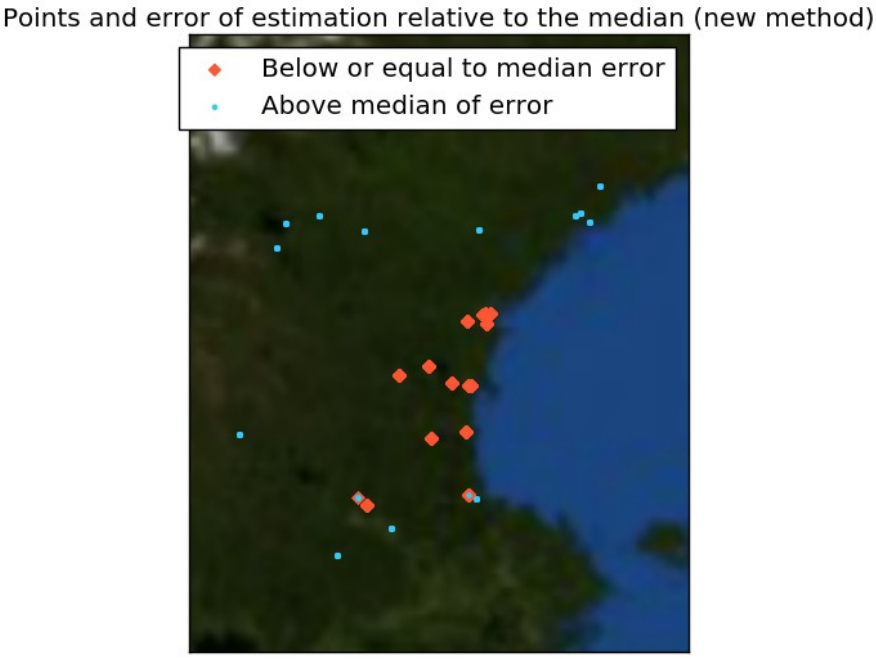

Figure 5.3: User location and indication of how large the error is when attempts are made to locate a user at that location with the new method

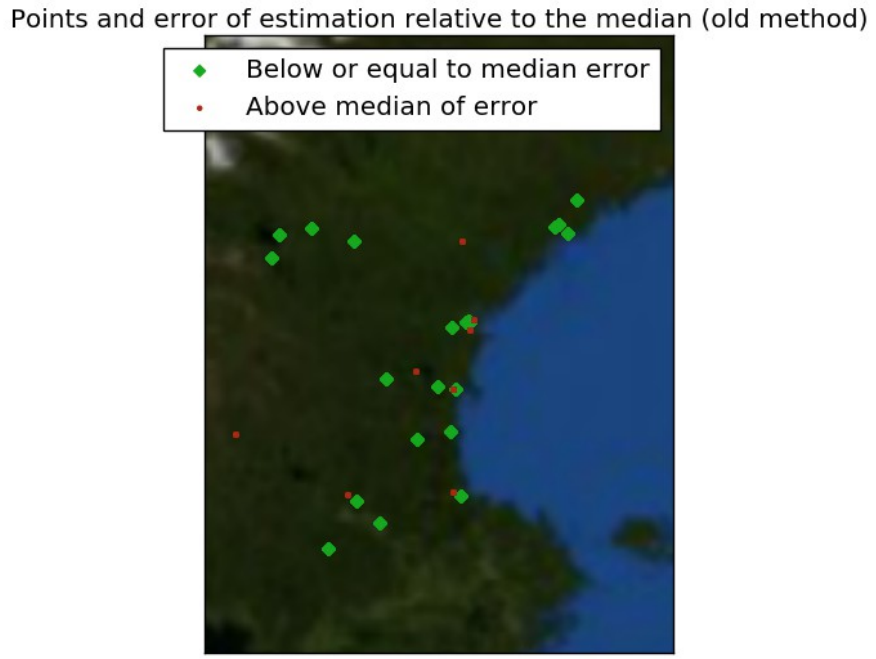

Figure 5.4: User location and indication of how large the error is when attempts are made to locate a user at that location with the old method

The new method has a below or equal to median error when attempting to locate users in the Sundsvall, Hudiksvall and Gävle regions while the larger errors usually occur well outside these regions, In the Örnsköldsvik region there is a cluster of users with above median errors of estimated locations, even though they are close to an urban area just like the users in the other urban regions whom have a below median error.

In figure 5.4 the same plot has been made but in this case the estimated locations from the old method. The pattern is not as clear as in the new method but there is an overrepresentation of below median estimations further inland in the more sparsely populated areas of the Mid Sweden region. There are above median estimations in the urban areas, especially Sundsvall, but there are no below median estimations in the Örnsköldsvik area in contrast to the new method.

6Conclusions

Without filtering the newer method is mostly worse than the database driven method already in place, only the maximum error rate is better on the new version which uses the trace routes but most of the time the old method performs better. However, if we discard poor estimations, judged by the standard distance deviation, the new method performs better then the old most of the time and also has a better best case and worst case performance, along with a better median and mean. This suggests that a geolocation algorithm using nothing but shared hops of known landmarks can be as viable as current, database driven methods.

The distributions of the two methods differed significantly, with or without the filter applied. The current method was skewed to smaller errors but also had a long tail, with very large worst errors recorded. The new method had a very uniform distribution but encompassed a smaller range than the current method. This suggests that the strength of the new method lies in its low worst case scenarios, probably because It uses locations of other users within the same region to estimate an unknown users location instead of router placement, which suggests that the position of other hosts in a geographic region is useful information.

The cross-validation procedure of this study chosen

This is the case of one VP and the original CBG algorithm saw a better performance when more VPs were implemented, but the increase was logarithmic meaning that the gain started to level out after a certain number of them were introduced [1]. This suggests that the performance with respect to VPs follow a logarithmic curve, meaning that there is a number of VPs results in diminishing returns for the performance. In this particular example, more VPs could discover more detours on a traceroute that could throw the algorithm and they could also see if there are any regional differences in the traceroutes that are significant, meaning that two landmarks might have the same traceroute from one VP but the pair of landmarks might have mostly different traceroutes from another, meaning they could offer clues as to wether or not these two landmarks are sufficiently "close". Another study on how different traceroutes

from different VPs to landmarks correlate with physical distance between the landmarks is recommended.

For the future, I suggest similar tests are carried out with more VPs than one. The landmark locations were also not entirely known, but rather assumed to be the address of the user associated with them. One future improvement could be to use IP-addresses associated with GPS outfitted devices to get better data. More users should also be used. Although 5000 rows are a lot, they only contain information about 80 users. Since the sample size is so small it's possible for it to become more skewed, leading to overfitting. Since each user was selected randomly, with each user having the probability of 67% of being selected, the proportions of the two sets was not entirely desired (two thirds for training and one third for testing) and the central limit theorem states that the more observations of the random selection process would follow the distribution more closely.

It is not known if more data would mean better or worse performance since there could be more scattering around the hops then what was observed, but there could also be hops with higher densities of users. The area of Mid Sweden is more than a quarter of Sweden, covering hundreds of thousands of square kilometers, but has very few urban areas so it would be interesting to see how more data would effect the algorithm.

After the final application of the filter, only three users remain from the original 29. This is a 90% decrease of the original dataset which means that in the vast majority of cases, the old method was better than the new method. The unreliable estimations of the new method could be handled in a variety of ways, such as defaulting to the old method and having both methods work in tandem. If the old method is to be properly phased out, however, the unreliable estimations could be marked as such and Mittmedia could offer them ads relevant for the entire Mid Sweden region. However, in light of the poor performance of the new method under these circumstances, it should be stressed that more VPs and better estimations of the physical position of IP-addresses should be tested to properly judge the method.

The possibility of cycles requires an algorithm with a very high complexity of $O(n^3)$ to resolve it. This is an opportunity for optimization, studies should be done on the cycles to see if they occur in clusters and if these clusters can be abstracted to a single node, but not jeopardize the information contained in the

traceroute. Then the algorithm to handle cycles could be scrapped all together since there is no possibility of cycles and the tree model can be fully abused. The CBG algorithm used delay measurements and traceroutes to estimate an IP-address's physical location. In the tests performed, every single IP-address did not respond to the ICMP packets which is suspected to because of firewalls. There are better tracerouting methods available, such as TCP-tracerouting which uses the TCP (Transport Control Protocol) protocol. The TCP protocol is vital to Internet communication between hosts meaning that there is a possibility for more tolerance from ISPs and routers. A firewall can still filter ports, but if there is no SYN/ACK received on the port another port could be tried to decide wether or not there really is a firewall filtering the packets.

6.1 Policy of user data

User data containing their location is sensitive information and from the start the users have been informed what happens to their data in compliance with the Swedish law PUL, and also have the ability to opt out. Using it to find the location of a user who has not willingly submitted their location could also be abused, meaning that they would need to be informed as they use the service that their location is being estimated and that they can opt out of it. What the service later does with this information has to be something the user is comfortable with and valuable to her. For example, maybe suggesting that there is a sale at her local store but not using it to make her a target for new advertising strategies. The algorithm is dependent on reliable land marks, meaning that if enough users opt out of submitting their address, the error rate could make it unusable meaning that if the users organize themselves in some way, and can opt out en masse, there could be checks and balances to make sure it is not abused.

References

[1]Z, Hu, J, Heidemann, Y, Pradkin, "Towards Geolocation of Millions of IP Addresses", *USC/ISI Technical Report ISI-TR-680,* 2012

[2]C, Bauer, "On the (In-)Accuracy of GPS Measures of Smartphones", *Conference: 11th International Conference on Advances in Mobile Computing & Multimedia (MoMM2013), At Vienna, Austria,* vol 11, 2013

[3]J, Bendale, J, Ratanaraj Kumar, "Review of Different IP Geolocation Methods and Concepts", *International Journal of Computer Science & Information Technology,* vol 5, 2014, pp. 436

[4]D, Komosny, M, Voznak, S, Bezzateev, K, Ganeshan, "The Use of European Internet Communication Properties for IP Geolocation", *Information Technology and Control,* nr. 1, 2016, pp. 77-85

[5]B, Gueye, A, Ziviani, M, Crovella, S, Fdida, "Constraint-based Geolocation of Internet Hosts", *IEEE Transactions on Networking*, vol. 14, nr. 6, 2006, pp. 1219-1232

[6]Y, Wang, D, Burgener, M, Flores, A, Kuzmanovic, C, Huang "Towards Street-Level Client-Independent IP Geolocation", *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, 2011, pp. 365-379

[7]Trafikverket, http://www.trafikverket.se/om-oss/var-verksamhet/sa-har-jobbar-vi-med/fran-planering-till-byggande/sa-har-ska-vi-fordela-de-fem-miljarderna/region-mitt/
Retrieved: 2016-12-12

[8]Y, Shavitt, N, Zilberman, "A Geolocation Databases Study", *IEEE Journal on Selected Areas of Communications,* nr. 29, 2011, pp. 2044-2056

[9]WhoIs IP, http://www.whoisip.se
Retrieved: 2016-12-27

[10]Ip2Location, https://www.iplocation.net/
Retrieved: 2016-12-27

[11]P, Gill, Y, Ganjali, B, Wong, D, Lie, "Dude, where's that IP? Circumventing measurement-based IP geolocation", *Proceedings of the 19th USENIX conference on Security*, 2010, pp. 16

[12]B, Wong, I, Stoyanov, E, Gün Sirer, "Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts", *4th Symposium on*

Network Systems Design and Implementation, Cambridge, Massachussets, USA, 2007

[13]Cisco: Internet Protocol IP Multicast Technology, http://www.cisco.com/en/US/tech/tk828/tech_brief09186a00800a4415.html#wp17758
Retrieved: 2016-12-12

[14]C, Yaowen, "Comparison of Graph Databases and Relational Databases When Handling Large-Scale Social Data", *University of Saskatchewan,* 2016-09-20

[15]Titan - Distributed Graph Database, http://titan.thinkaurelius.com/
Retrieved: 2017-01-06

[16]Graph DB - by Ontotext, http://graphdb.ontotext.com/
Retrieved: 2017-01-06

[17]NoSQL - List of NoSQL Databases, http://nosql-database.org/
Retrieved: 2017-01-07

[18]DB-Engines Ranking, http://db-engines.com/en/ranking
Retrieved: 2017-01-06

[19]MongoDB - FAQ, https://www.mongodb.com/faq
Retrieved: 2017-01-06

[20]Stephen Pimentel - The Return of ACID in the Design of NoSQL Databases, http://www.methodsandtools.com/archive/acidnosqldatabase.php
Retrieved: 2017-01-10

[21]Movable-Type, http://www.movable-type.co.uk/scripts/gis-faq-5.1.html
Retrieved: 2016-12-12

[22]A, Sylvain, C, Alain, *A survey of cross-validation prcoedures for model selection, Statistics Surveys,* nr. 4, 2010, pp. 40-79

[23]RFC 1393 - Internet Engineering Task Force, https://tools.ietf.org/html/rfc1393
Retrieved: 2017-01-04

[24]Man page of Traceroute for Linux OS, https://linux.die.net/man/8/traceroute
Retrieved: 2017-01-04

[25]C, Vicknair, M, Macias, Z, Zhao, X, Nan, Y, Chen, D, Wilkins, "A comparison of a graph database and a relational database: a data provenance perspective", *ACM SE '10 Proceedings of the 48th annual Southeast Regional Conference*, 2010, art. nr. 42