Towards Street-Level Client-Independent IP Geolocation

Yong Wang
UESTC/Northwestern University

Daniel Burgener
Northwestern University

Marcel Flores
Northwestern University

Aleksandar Kuzmanovic
Northwestern University

Cheng Huang
Microsoft Research

Abstract

A highly accurate client-independent geolocation service stands to be an important goal for the Internet. Despite an extensive research effort and significant advances in this area, this goal has not yet been met. Motivated by the fact that the best results to date are achieved by utilizing additional 'hints' beyond inherently inaccurate delay-based measurements, we propose a novel geolocation method that fundamentally escalates the use of external information. In particular, many entities (*e.g.*, businesses, universities, institutions) host their Web services locally and provide their actual geographical location on their Websites. We demonstrate that the information provided in this way, when combined with network measurements, represents a precious geolocation resource. Our methodology automatically extracts, verifies, utilizes, and opportunistically inflates such Web-based information to achieve high accuracy. Moreover, it overcomes many of the fundamental inaccuracies encountered in the use of absolute delay measurements. We demonstrate that our system can geolocate IP addresses 50 *times* more accurately than the best previous system, *i.e.*, it achieves a median error distance of 690 meters on the corresponding data set.

1 Introduction

Determining the geographic location of an Internet host is valuable for a number of Internet applications. For example, it simplifies network management in large-scale systems, helps network diagnoses, and enables location-based advertising services [19,29]. While coarse-grained geolocation, *e.g.*, at the state- or city-level, is sufficient in a number of contexts [22], the need for a *highly accurate* and reliable geolocation service has been identified as an important goal for the Internet (*e.g.*, [11, 19]). Such a system would not only improve the performance of existing applications, but would enable the development of novel ones.

While client-assisted systems capable of providing highly accurate IP geolocation inferences do exist [3, 5, 9], many applications such as location-based access restrictions, context-aware security, and online advertising, can not rely on clients' support for geolocation. Hence, a highly accurate *client-independent* geolocation system stands to be an important goal for the Internet.

An example of an application that already extensively uses geolocation services, and would significantly benefit from a more accurate system, is online advertising. For example, knowing that a Web user is from New York is certainly useful, yet knowing the exact part of Manhattan where the user resides enables far more effective advertising, *e.g.*, of neighboring businesses. On the other side of the application spectrum, example services that would benefit from a highly accurate and dependable geolocation system, are the enforcement of location-based access restrictions and context-aware security [2]. Also of rising importance is cloud computing. In particular, in order to concurrently use public and private cloud implementations to increase scalability, availability, or energy efficiency (*e.g.*, [26]), a highly accurate geolocation system can help select a properly dispersed set of client-hosted nodes within a cloud.

Despite a decade of effort invested by the networking research community in this area, *e.g.*, [13, 16–20, 22, 24], and despite significant improvements achieved in recent years (*e.g.*, [19, 29]), the desired goal, a geolocation service that would actually enable the above applications, has not yet been met. On one hand, commercial databases currently provide rough and incomplete location information [19, 24]. On the other hand, the best result reported by the research community (to the best of our knowledge) was made by the Octant system [29]. This system was able to achieve a median estimation error of 22 miles (35 kilometers). While this is an admirable result, as we elaborate below, it is still insufficient for the above applications.

The key contribution of our paper lies in designing a novel client-independent geolocation methodology and in deploying a system capable of achieving highly accurate results. In particular, we demonstrate that our system can geolocate IP addresses with a median error distance of 690 meters in an academic environment. Comparing to recent results on the same dataset shows that we improve the median accuracy by 50 times relative to [29] and by approximately 100 times relative to [19]. Improvements at the tail of the distribution are even more significant.

Our methodology is based on the following two insights. First, many entities host their Web services locally. Moreover, such Websites often provide the actual geographical location of the entity (*e.g.*, business and university) in the form of a postal address. We demonstrate that the information provided in this way represents a *valuable* resource, *i.e.*, it provides access to a large number of highly accurate landmarks that we can exploit to achieve equally accurate geolocation results. We thus develop a methodology that effectively mines, verifies, and utilizes such information from the Web.

Second, while we utilize absolute network delay measurements to estimate the coarse-grained area where an IP is located, we argue that absolute network delay measurements are fundamentally limited in their ability to achieve fine-grained geolocation results. This is true in general even when additional information, *e.g.*, network topology [19] or negative constraints such as uninhabitable areas [29], is used. One of our key findings, however, is that *relative network delays* still heavily correlate with geographical distances. We thus fully abandon the use of absolute network delays in the final step of our approach, and show that a simple method that utilizes only relative network distances achieves the desired accuracy.

Combining these two insights into a single methodology, we design a three-tier system which begins at the large, course-grained scale, first tier where we utilize a distance constraint-based method to geolocate a target IP into an area. At the second tier, we effectively utilize a large number of Web-based landmarks to geolocate the target IP into a much smaller area. At the third tier, we opportunistically inflate the number of Web landmarks and demonstrate that a simple, yet powerful, closest node selection method brings remarkably accurate results.

We extensively evaluate our approach on three distinct datasets – Planetlab, residential, and an online maps dataset – which enables us to understand how our approach performs on an academic network, a residential network, and in the wild. We demonstrate that our algorithm functions well in all three environments, and that it is able to locate IP addresses in the real world with high accuracy. The median error distances for the three sets are 0.69 km, 2.25 km, and 2.11 km, respectively.

We demonstrate that factors that influence our system's accuracy are: (i) Landmark density, *i.e.*, the more landmarks there are in the vicinity of the target, the better accuracy we achieve. (ii) Population density, *i.e.*, the more people live in the vicinity of the target, the higher probability we obtain more landmarks, the better accuracy we achieve. (iii) The AS factor, *i.e.*, under the same landmark density circumstances, the landmarks from the same AS bring more accurate results. (iv) Access technology, *i.e.*, our system has slightly reduced accuracy (by approximately 700 meters) for cable users relative to DSL users. While our methodology effectively resolves the last mile delay inflation problem, it is necessarily less resilient to the high last-mile latency *variance*, common for cable networks.

Given that our approach utilizes Web-based landmark discovery and network measurements on the fly, one might expect that the measurement overhead (crawling in particular) hinders its ability to operate in real time. We show that this is not the case. In a fully operational network measurement scenario, all the measurements could be done within 1-2 seconds. Indeed, Web-based landmarks are stable, reliable, and long lasting resources. Once discovered and recorded, they can be reused for many measurements and re-verified over longer time scales.

2 A Three-Tier Methodology

Our overall methodology consists of two major components. The first part is a three-tier active measurement methodology. The second part is a methodology for extracting and verifying accurate Web-based landmarks. The geolocation accuracy of the first part fundamentally depends on the second. For clarity of presentation, in this section we present the three-tier methodology by simply assuming the existence of Web-based landmarks. In the next section, we provide details about the extraction and verification of such landmarks.

We deploy the three-tier methodology using a distributed infrastructure. Motivated by the observation that the sparse placement of probing vantage points can avoid gathering redundant data [31], we collect 163 publicly available ping and 136 traceroute servers geographically dispersed at major cities and universities in the US.

2.1 Tier 1

Our final goal is to achieve a high level of geolocation precision. We achieve this goal gradually, in three steps, by incrementally increasing the precision in each step. The goal of the first step is to determine a coarse-grained region where the targeted IP is located. In an attempt

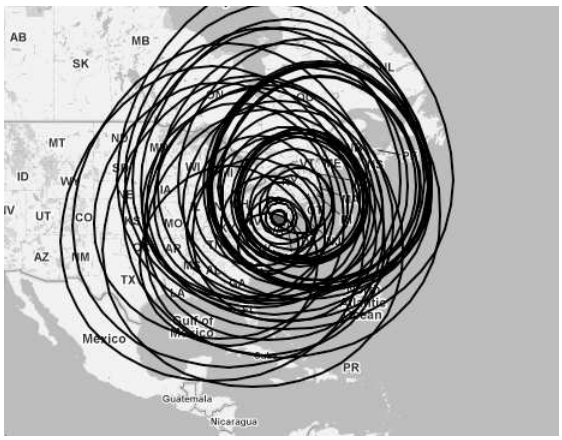

Figure 1: An example of intersection created by distance constraints

not to 'reinvent the wheel,' we use a variant of a well established constrained-based geolocation (CBG) method [17], with minor modifications.

To geolocate the region of an IP address, we first send probes to the target from the ping servers, and convert the delay between each ping server and the target into a geographical distance. Prior work has shown that packets travel in fiber optic cables at 2/3 the speed of light in a vacuum (denoted by c) [23]. However, others have demonstrated that 2/3 c is a loose upper bound in practice due to transmission delay, queuing delay *etc.* [17, 19]. Based on this observation, we adopt 4/9 c from [19] as the converting factor between measured delay and geographical distance. We also demonstrate in Section 4, by using this converting factor, we are always capable of yielding a viable area covering the targeted IP.

Once we establish the distance from each vantage point, *i.e.*, ping server, to the target, we use multilateration to build an intersection that covers the target using known locations of these servers. In particular, for each vantage point, we draw a ring centered at the vantage point, with a radius of the measured distance between the vantage point and the target. As we show in Section 4, this approach indeed allows us to always find a region that covers the targeted IP.

Figure 1 illustrates an example. It geolocates a collected target (we will elaborate the way of collecting the targets in the wild in Section 4.1.2) whose IP address is 38.100.25.196 and whose postal address is '1850, K Street NW, Washington DC, DC, 20006'. We draw rings centered at the locations of our vantage points. The radius of each ring is determined by the measured distance between the vantage point (the center of this ring) and the target. Finally, we geolocate this IP in an area indicated by the shaded region, which covers the target, as shown in Figure 1.

Thus, by applying the CBG approach, we manage to geolocate a region where the targeted IP resides. According to [19, 29], CBG achieves a median error between 143 km and 228 km distance to the target. Since we strive for a much higher accuracy, this is only the starting point for our approach. To that end, we depart from pure delay measurements and turn to the use of external information available on the Web. Our next goal is to further determine a subset of ZIP Codes, *i.e.*, smaller regions that belong to the bigger region found via the CBG approach. Once we find the set of ZIP Codes, we will search for additional websites served within them. Our goal is to extract and verify the location information about these locally-hosted Web services. In this way, we obtain a number of accurate Web-based landmarks that we will use in Tiers 2 and 3 to achieve high geolocation accuracy.

To find a subset of ZIP Codes that belong to the given region, we proceed as follows. We first determine the centroid of the intersection area. Then, we draw a ring centered in the intersection centroid with a diameter of 5 km. Next, we sample 10 latitude and longitude pairs at the perimeter of this ring, by rotating by 36 degrees between each point. For the 10 initial points, we verify that they belong to the intersection area as follows. Denote by U the set of latitude and longitude pairs to be verified. Next, denote by V the set of all vantage points, *i.e.*, ping servers, with known location. Each vantage point v_i is associated with the measured distance between itself and the target, denoted by r_i. We wish to find all $u \in U$ that satisfy

$$\text{distance}(u, v_i) \leq r_i \text{ for all } v_i \in V$$

The distance function here is the great-circle distance [27], which takes into account the earth's sphericity and is the shortest distance between any two points on the surface of the earth measured along a path on the surface of the earth. We repeat this procedure by further obtaining 10 additional points by increasing the distance from the intersection centroid by 5 km in each round (*i.e.*, to 10 km in the second round, 15 km in the third *etc.*). The procedure stops when not a single point in a round belongs to the intersection. In this way, we obtain a sample of points from the intersection, which we convert to ZIP Codes using publicly available services [4]. Thus, with the set of ZIP Codes belonging to the intersection, we proceed to Tier 2.

2.2 Tier 2

Here, we attempt to further reduce the possible region where the targeted IP is located. To that end, we aim to find Web-based landmarks that can help us achieve this goal. We explain the methodology for obtaining such

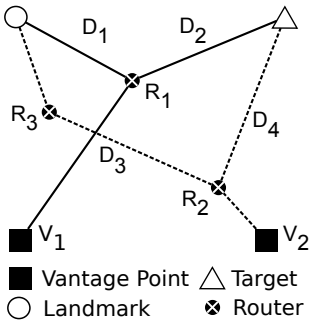

Figure 2: An example of measuring the delay between landmark and target

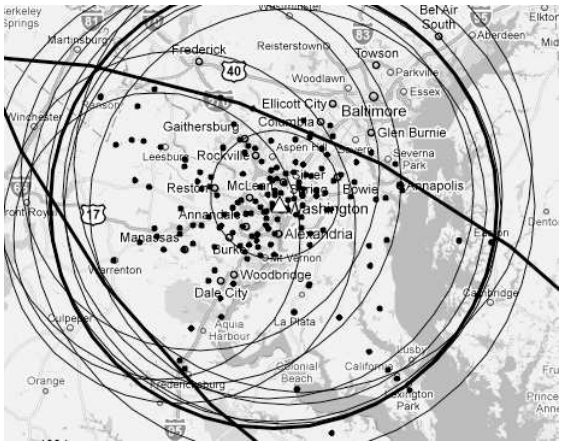

Figure 3: An example of shrinking the intersection

landmarks in Section 3. Although these landmarks are passive, *i.e.*, we cannot actively send probes to other Internet hosts using them, we use the traceroute program to *indirectly* estimate the delay between landmarks and the target.

Learning from [10] that the more traceroute servers we use, the more direct a path between a landmark and the target we can find, we first send traceroute probes to the landmark (the empty circle in Figure 2) and the target (the triangle in Figure 2) from all traceroute servers (the solid squares V_1 and V_2 in Figure 2). For each vantage point, we then find the closest common router to the landmark and the target, shown as R_1 and R_2 in Figure 2, on the routes towards both the landmark and the target. Next, we calculate the latency between the common router and the landmark (D_1 and D_3 in Figure 2) and the latency between the common router and the target (D_2 and D_4 in Figure 2). We finally select the sum (D) of two latencies as the delay between landmark and target. In the example above, from V_1's point of view, the delay D between the target and the landmark is $D = D_1 + D_2$, while from V_2's perspective, the delay D is $D = D_3 + D_4$.

Since different traceroute servers have different routes to the destination, the common routers are not necessarily the same for all traceroute servers. Thus, each vantage point (a traceroute server) can estimate a different delay between a Web-based landmark and the target. In this situation, we choose the minimum delay from all traceroute servers' measurements as the final estimation of the latency between the landmark and the target. In Figure 2, since the path between landmark and target from V_1's perspective is more direct than that from V_2's ($D_1 + D_2 < D_3 + D_4$), we will consider the sum of D_1 and D_2 ($D_1 + D_2$) as the final estimation.

Routers in the Internet may postpone responses. Consequently, if the delay on the common router is inflated, we may underestimate the delay between landmark and target. To examine the 'quality' of the common router we use, we first traceroute different landmarks we collected previously and record the paths between any two landmarks, which also branch at that router. We then calculate the great circle distance [27] between two landmarks and compare it with their measured distance. If we observe that the measured distance is smaller than the calculated great circle distance for any pair of landmarks, we label this router as 'inflating', record this information, and do not consider its path (and the corresponding delay) for this or any other measurement.

Through this process, we can guarantee that the estimated delay between a landmark and the target is not underestimated. Nonetheless, such estimated delay, while converging towards the real latency between the two entities, is still usually larger. Hence, it can be considered as the upper bound of the actual latency. Using multilateration with the upper bound of the distance constraints, we further reduce the feasible region using the new tier 2 and the old tier 1 constraints.

Figure 3 shows the zoomed-in subset of the constrained region together with old tier 1 constraints, marked by thick lines, and new tier 2 constraints, marked by thin lines. The figure shows a subset of sampled landmarks, marked by the solid dots, and the IP that we aim to geolocate, marked by a triangle. The tier 1 constrained area contains 257 distinctive ZIP Codes, in which we are able to locate and verify 930 Web-based landmarks. In the figure, we show only a subset of 161 landmarks for a clearer presentation. Some sampled landmarks lie outside the original tier 1 level intersection. This happens because the sampled ZIP Codes that we discover at the borders of the original intersection area typically spread outside the intersection as well. Finally, the figure shows that the tier 2 constrained area is approximately one order of magnitude smaller than the original tier 1 area.

2.3 Tier 3

In this final step, our goal is to complete our geolocation of the targeted IP address. We start from the region constrained in Tier 2, and aim to find all ZIP Codes in this region. To this end, we repeat the sampling procedure deployed in the Tier 2. This time from the centroid of the Tier 2 constrained intersection area, and at a higher granularity. In particular, we extend the radius distance by 1 km in each step, and apply a rotation angle of 10 degrees. Thus, we achieve 36 points in each round. We apply the same stopping criteria, *i.e.*, when no points in a round belong to the intersection. This finer-grain sampling process enables us to discover all ZIP Codes in the intersection area. For ZIP Codes that were not found in the previous step, we repeat the landmark discovery process (Section 3). Moreover, to obtain the distance estimations between newly discovered landmarks and the target, we apply the active probing traceroute process explained above.

Finally, knowing the locations of all Web-based landmarks and their estimated distances to the target, we select the landmark with the minimum distance to the target, and associate the target's location with it. While this approach may appear ad hoc, it signifies one of the key contributions of our paper. We find that on the smaller-scale, *relative distances* are preserved by delay measurements, overcoming many of fundamental inaccuracies encountered in the use of absolute measurements. For example, a delay of several milliseconds, commonly seen at the last mile, could place an estimate of a scheme that relies on absolute delay measurements hundreds of kilometers away from the target. On the contrary, selecting the closest node in an area densely populated with landmarks achieves remarkably accurate estimates, as we show below in our example case, and demonstrate systematically in Section 4 via large-scale analysis.

Figure 4 shows the striking accuracy of this approach. We manage to associate the targeted IP location with a landmark which is 'across the street', *i.e.*, only 0.103 km distant from the target. We analyze this result in more detail below. Here, we provide the general statistics for the Tier 3 geolocation process. In this last step, we discover 26 additional ZIP Codes and 203 additional landmarks in the smaller Tier 2 intersection area. We then associate the landmark, which is at '1776 K Street Northwest, Washington, DC' and has a measured distance of 10.6 km, yet a real geographical distance of 0.103 km, with the target. To clearly show the association, Figure 4 zooms into a very finer-grain street level in which the constrained rings and relatively more distant landmarks are not shown.

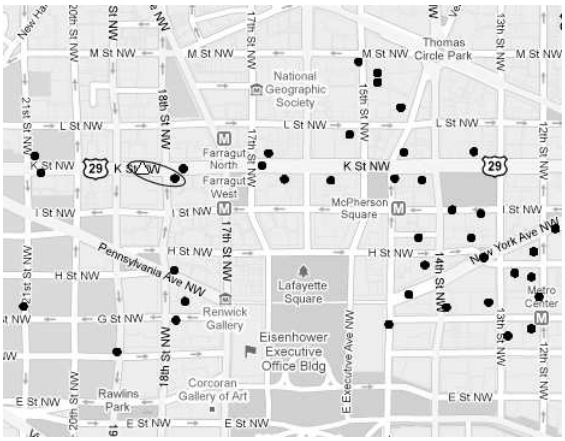

Figure 4: An example of associating a landmark with the target as the result

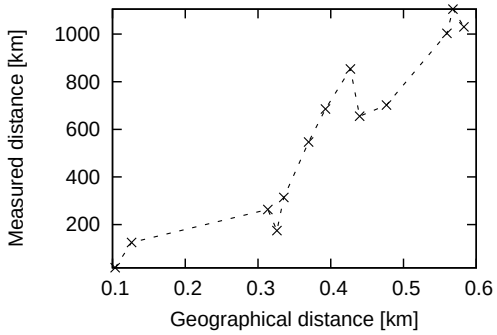

Figure 5: Measured distance vs. geographical distance.

2.3.1 The Power of Relative Network Distance

Here, we explore how the relative network distance approach achieves such good results. Figure 5 sheds more light on this phenomenon. We examine the 13 landmarks within 0.6 km of the target shown in Figure 4. For each landmark, we plot the distance between the target and the Web-based landmarks (y-axis) (measured via the traceroute approach) as a function of the actual geographical distance between the landmarks and the target (x-axis). The first insight from the figure is that there is indeed a significant difference between measured distance, *i.e.*, their upper bounds, and the real distances. This is not a surprise. A path between a landmark, over the common router, to the destination (Figure 2) can often be circuitous and inflated by queuing and processing delays, as demonstrated in [19]. Hence, the estimated distance dramatically exceeds the real distance, by approximately three orders of magnitude in this case.

However, Figure 5 shows that the distance estimated via network measurements (y-axis) is largely *in propor-*

tion with the actual geographical distance. Thus, despite the fact that the direct relationship between the real geographic distance and estimated distance is inevitably lost in inflated network delay measurements, the relative distance is largely preserved. This is because the network paths that are used to estimate the distance between landmarks and the target share vastly common links, hence experience similar transmission- and queuing-delay properties. Thus, selecting a landmark with the smallest delay is an effective approach, as we also demonstrate later in the text.

To further analyze the results shown in Figure 5 we look at the AS-level properties of the target and the associated landmarks. Among the 13 landmarks within 0.6 km, 12 of them are in different ASes than the target, and only one (the closest one to the target) is in the same AS as the target. But the striking insight is that even if the closest landmark from the same AS did not exist, we still would have selected a good landmark, only 0.125 km distant from the target. Indeed, the other points in the figure show that the relative order of the closest landmarks to the target is largely preserved.

We systematically evaluate this phenomenon in Section 4. Below, we explain how to automatically extract and verify Web-based landmarks which are proven to be a valuable geolocation resource.

3 Extracting and Verifying Web-Based Landmarks

Many entities, *e.g.*, companies, academic institutions, and government offices, host their Web services locally. One implication of this setup is that the actual geographic addresses, (in the form of a street address, city, and ZIP Code), which are typically available at companies' and universities' home Web pages, correspond to the actual physical locations where these services are located. Accordingly, the geographical location of the corresponding web-servers' IP addresses becomes available, and the servers themselves become viable geolocation landmarks. Indeed, we have demonstrated above that such Web-based landmarks constitute an important geolocation resource. In this section, we provide a comprehensive methodology to automatically extract and verify such landmarks.

3.1 Extracting Landmarks

To automatically extract landmarks, we mine numerous publicly available mapping services. In this way, we are able to associate an entity's postal address with its domain name using such mapping services, which have already thoroughly and carefully established this association. Note that the use of online mapping services is a convenience, *not* a requirement for our approach. Indeed, the key resource that our approach relies upon is the existence of geographical addresses at locally hosted websites, which can be accessed directly. We discuss this issue later in Section 5.

In order to discover landmarks in a given ZIP Code, which is an important primitive of our methodology explained in Section 2 above, we proceed as follows. We first query the mapping service by a request that consists of the desired ZIP Code and a keyword, *i.e.*, either 'business', 'university', and 'government office'. The service replies with a list of companies, academic institutions, or government offices within, or close to, this ZIP Code. Each landmark in the list includes the geographical location of this entity at the street-level precision and its web site's domain name.

As an example, a jewelry company at '55 West 47th Street, Manhattan, New York, NY, 10036', with the domain name www.zaktools.com, is a landmark for the ZIP Code 10036. For each entity, we also convert its domain name into an IP address to form a (domain name, IP address, and postal address) mapping. For the example above, the mapping in this case is (www.zaktools.com, 69.33.128.114, '55 West 47th Street, Manhattan, New York, NY, 10036'). A domain name can be mapped into several IP addresses. Initially, we map each of the IP addresses to the same domain name and postal address. Then, we verify all the extracted IP addresses using the methodology we present below.

3.2 Verifying Landmarks

A geographic address extracted from a Web page using the above approach may not correspond to the associated server's physical address for several reasons. Below, we explain such scenarios and propose verification methods to automatically detect and remove such landmarks.

3.2.1 Address Verification

The businesses and universities provided by online mapping services may be the landmarks *near* the areas covered by the ZIP Code, not necessarily *within* the ZIP Code. Thus, we first examine the ZIP Code in the postal address of each landmark. If a landmark has a ZIP Code different from the one we searched for, we remove it from the list of candidate landmarks. For example, for the ZIP Code 10036, a financial services company called Credit Suisse (www.credit-suisse.com) at '11 Madison Ave, New York, NY, 10010' is returned by Google Maps as an entity near the specified ZIP Code 10036. Using our verification procedure, we remove such a landmark from the list of landmarks associated with the 10036 ZIP Code.

3.2.2 Shared Hosting and CDN Verification

Additionally, a company may not always host its website locally. It may utilize either a CDN network to distribute its content or use shared hosting techniques to store its archives. In such situations, there is no one-to-one mapping between an IP address and a postal address in both CDN network and shared hosting cases. In particular, a CDN server may serve multiple companies' websites with distinct postal addresses. Likewise, in the shared hosting case a single IP address can be used by hundreds or thousands of domain names with diverse postal addresses. Therefore, for a landmark with such characteristics, we should certainly not associate its geographical location with its domain name, and in turn its IP address. On the contrary, if an IP address is solely used by a single entity, the postal address is much more trustworthy. While not necessarily comprehensive, we demonstrate that this method is quite effective, yet additional verifications are needed, as we explain in Section 3.2.3 below.

In order to eliminate a bad landmark, we access its website using (i) its domain name and (ii) its IP address independently. If the contents, or heads (distinguished by <head> and </head>), or titles (distinguished by <title> and </title>) returned by the two methods are the same, we confirm that this IP address belongs to a single entity. One complication is that if the first request does not hit the '$final$' content, but a redirection, we will extract the '$real$' URL and send an additional request to fetch the '$final$' content.

Take the landmark (www.manhattanmailboxes.com) at '676A 9 Avenue, New York, NY, 10036' as an example. We end up with a web page showing 'access error' when we access this website via its IP address, 216.39.57.104. Indeed, searching an online shared hosting check [8], we discover that there are more than 2,000 websites behind this IP address.

3.2.3 The Multi-Branch Verification

One final scenario occurs often in the real world: A company headquartered in a place where its server is also deployed may open a number of branches nationwide. Likewise, a medium size organization can also have its branch offices deployed locally in its vicinity. Each such branch office typically has a different location in a different ZIP Code. Still, all such entities have the same domain name and associated IP addresses as their headquarters.

As we explained in Section 2, we retrieve landmarks in a region covering a number of ZIP Codes. If we observe that some landmarks, with the same domain name, have different locations in different ZIP Codes, we remove them all. For example, the Allstate Insurance Company, with the domain name 'www.allstate.com' has many affiliated branch offices nationwide. As a result, it shows up multiple times for different ZIP Codes in an intersection. Using the described method, we manage to eliminate all such occurrences.

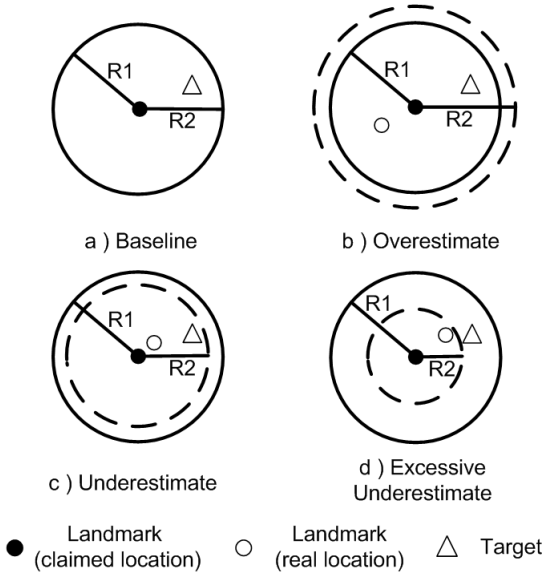

Figure 6: The effects of improper landmark

3.3 Resilience to Errors

Applying the above methods, we can remove the vast majority of erroneous Web landmarks. However, exceptions certainly exist. One example is an entity (*e.g.*, a company) without any branch offices that hosts a website used exclusively by that company, but does not locate its Web server at the physical address available on the Website. In this case, binding the IP address with the given geographical location is incorrect, hence such landmarks may generate errors. Here, we evaluate the impact that such errors can have on our method's accuracy. Counterintuitively, we show that the larger the error distance is between the claimed location (the street-level address on a website) and the real landmark location, the more resilient our method becomes to such errors. In all cases, we demonstrate that our method poses significant resilience to false landmark location information.

Figure 6 illustrates four possible cases for the relationship between a landmark's real and claimed location. The figure denotes the landmark's real location by an empty circle, the landmark's claimed location by a solid circle, and the target by a triangle. Furthermore, denote $R1$ as the claimed distance, *i.e.*, the distance between the claimed location and the target. Finally, denote $R2$ as the measured distance between the landmark's actual location and the target.

Figure 6(a) shows the baseline error-free scenario. In this case, the claimed and the real locations are identical. Hence, $R1 = R2$. Thus, we can draw a ring that is centered at the solid circle and is always able to contain the target, since the upper bound is used to measure the distance in Section 2.2.

Figure 6(b) shows the case when the claimed landmark's location is different from the real location. Still, the real landmark is farther away from the target than the claimed location is, as shown in the figure. Hence, $R2 > R1$. Thus, we will draw a bigger ring with the radius of $R2$, shown as the dashed curve, than the normal case with the radius of $R1$. Thus, such an overestimate yields a larger coverage that always includes the target. Hence, our algorithm is unharmed, since the target remains in the feasible region.

Figures 6 (c) and (d) show the scenario when the real landmark's location is closer to the target than the claimed location is, *i.e.*, $R2 < R1$. There are two sub scenarios here. In the underestimate case (shown in Figure 6(c)), the real landmark location is slightly closer to the target and the measured delay is only a little smaller than it should be. However, since the upper bound is used to measure the delay and convert it into distance, such underestimates can be counteracted. Therefore, we can still draw a ring with a radius of $R2$, indicated by the dashed curve, covering the target. In this case, the underestimate does not hurt the geolocation process.

Finally, in the excessive underestimate case (shown in Figure 6), the landmark is actually quite close to the target and the measured delay is much smaller than expected. Consequently, we end with a dashed curve with the radius of $R2$ that does not include the target, even when the upper bounds are considered. In this case, the excessive underestimate leads us to an incorrect intersection or an improper association between the landmark and the target ($R2 < R1$).

The error resilience proof. While it is intuitive that the excessive underestimate case is not likely to happen, here we offer a proof. We develop a simple, yet illustrative model, to demonstrate that this is indeed the case. To simplify the presentation, we ignore the upper bound effects discussed above, which can only improve our situation, and assume the model shown in Figure 7. The triangle again denotes the target that we want to geolocate, the solid circle is the location where the landmark claims to be, and the empty circle represents landmark's actual location. Denote R as the claimed distance to the target, and D as the distance between the landmark's claimed location and its actual location, as shown in the figure.

Consider the case when $D > 2R$, *i.e.*, the landmark's real location is beyond the light gray circle shown in Figure 7. In that case, the distance from the landmark's real location to the target will be $> R$. Thus, it will therefore only provide a harmless overestimate. The overestimate is harmless because a ring of radius $> R$ centered at the landmarks claimed location will still cover the target. In general, for a given D, the probability P that the point will not cause harm is given by the percentage of the perimeter of the circle of radius D centered at the landmark's claimed location that lies within the light gray shaded area. If we consider when $2R \geq D \geq R$, we see that this probability is $1 > P \geq 2/3$. If $D < R$, we see that we will have $2/3 > P \geq 1/2$, with $1/2$ occurring when the claimed and actual locations overlap. We therefore see that the probability that a landmark will be harmless, even if it is closer than claimed, is always $\geq 1/2$. A proof of the formula used to calculate these probabilities can be found in the Appendix.

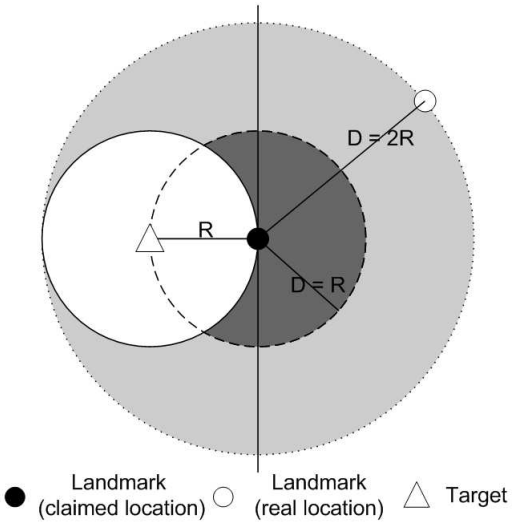

Figure 7: The probability is always larger than 0.5

The fact that the distance between a target and a landmark's *claimed* location is small by design, *e.g.*, in the same ZIP Code, implies that the claimed distance R is usually small. Consequently, even a moderate distance between the claimed and actual landmark's location, D, already puts us in the safe region of $P = 1$. As we demonstrate in theory above and in practice at Section 4, our algorithm is tolerant to such landmarks in most cases and their existence does not cause harm.

4 Evaluation

4.1 Datasets

We use three different datasets, Planetlab, residential, and online maps, as we explain below. Comparing with the large online maps dataset, the number of targets in the Planetlab and the residential datasets are relatively small. However, these two datasets help us gain valuable

insights about the performance of our method in different environments, since the online maps dataset can contain both types of targets.

4.1.1 Planetlab dataset

One method commonly used to evaluate the accuracy of IP geolocation systems is to geolocate Planetlab nodes, *e.g.*, [19, 29]. Since the locations of these nodes are known publicly (universities must report the locations of their nodes), it is straightforward to compare the location given by our system with the location provided by the Planetlab database. We select 88 nodes from Planetlab, limiting ourselves to at most one node per location. Others (*e.g.*, [19]) have observed errors in the given Planetlab locations. Thus, we manually verify all of the nodes locations.

4.1.2 Residential dataset

Since the set of Planetlab nodes are all located on academic networks, we needed to validate our approach on residential networks as well. Indeed, many primary applications of IP geolocation target users on residential networks. In order to do this, we created a website, which we made available to our social networks, widely dispersed all over the US. The site automatically records users' IP addresses and enables them to enter their postal address and the access provider. In particular, we enable six selections for the provider: AT&T, Comcast, Verizon, other ISPs, University, and Unknown. Moreover, we explicitly request that users not enter their postal address if they are accessing this website via proxy, VPN, or if they are unsure about their connection. We then distribute the link to many people via our social networks, and obtained 231 IP address and location pairs.

Next, we eliminate duplicate IPs, 'dead' IPs that are not accessible over the course of the experiment, which is one-month after the data was collected. We also eliminate a large number of IPs with access method 'university' or 'unknown', since we intend to extract residential IPs and compare with those of academic IPs in Section 4.2. After elimination, we are left with 72 IPs.

4.1.3 Online Maps dataset

We obtained a large-scale query trace from a popular online maps service. This dataset contains three-months of users' search logs for driving directions.[1] Each record consists of the user access IP address, local access time at user side, user browser agent, and the driving sequence represented by two pairs of latitude and longitude points.

[1] We respect a request of this online map service company and do not disclose the number of requests and collected IPs here and in the rest of the paper.

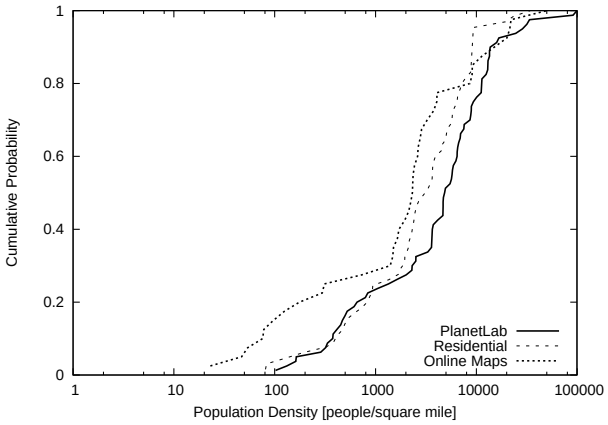

Figure 8: The distribution of the population density of three datasets

Our hypothesis here is that if we observe a location, as either source or destination in the driving sequence, periodically associated with an IP address, then this IP address is likely at that location. To extract such association from the dataset, we employ a series of strict heuristics as follows.

We first exclude IP addresses associated with multiple browser agents. This is because it is unclear whether this IP address is used by only one user with multiple browsers or by different users. We then select IP addresses for which a single location appears at least four times in each of the three months, since such IP addresses with 'stable' search records are more likely to provide accurate geolocation information than the ones with only a few search records. We further remove IP addresses that are associated with two or more locations that appear at least four times. Finally we remove all 'dead' IPs from the remaining dataset.

4.1.4 Dataset characteristics

Here, our goal is to explore the characteristics of the locations where the IP addresses of the three datasets are. In particular, population density is an important parameter that indicates the rural vs. urban nature of the area in which an IP address resides. We will demonstrate below that this parameter influences the performance of our method, since urban areas typically have a large number of web-based landmarks.

Figure 8 shows the distribution of the population density of the ZIP Code at which the IP addresses of the three datasets locate. We obtain the population density for each ZIP Code by querying the website City Data [1]. Figure 8 shows that our three datasets cover both rural areas, where the population density is small, and urban areas, where the population density is large. In particu-

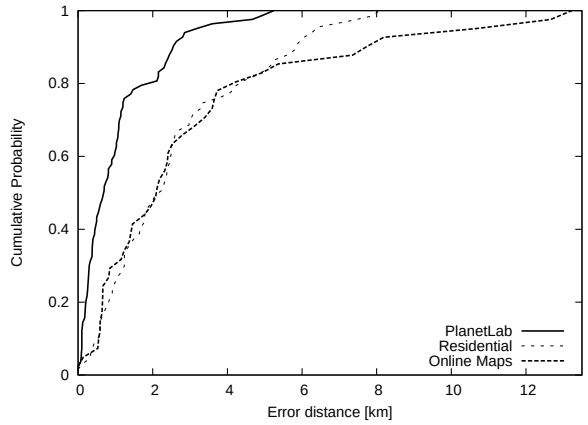

Figure 9: Comparison of error distances of three datasets

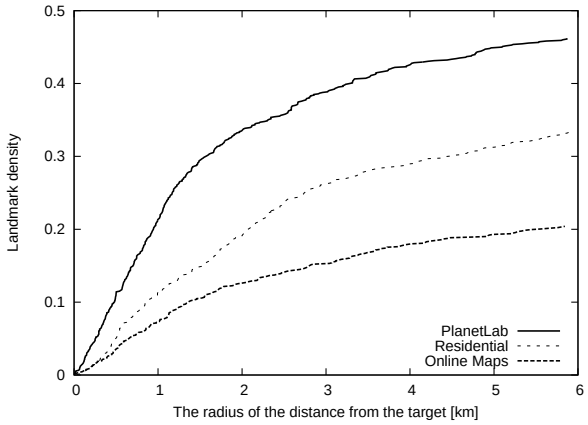

Figure 10: Landmark density of three datasets

lar, all three datasets have more than 20% of IPs in ZIP Codes whose population density is less than 1,000. The figure also shows that PlanetLab dataset is the most 'urban' one, while the Online Maps datasets has the longest presence in rural areas. In particular, about 18% of IPs in the Online Maps dataset reside in ZIP Codes whose population density is less than 100.

4.2 Experimental results

4.2.1 Baseline results

Figure 9 shows the results for the three datasets. In particular, it depicts the cumulative probability of the error distance, *i.e.*, the distance between a target's real location and the one geolocated by our system. Thus, the closer the curve is to the upper left corner, the smaller the error distance, and the better the results. The median error for the three datasets, a measure typically used to represent the accuracy of geolocation systems [17, 19, 29], are 0.69 km for Planetlab, 2.25 km for the residential dataset, and 2.11 km for the online maps dataset. Beyond excellent median results, the figure shows that the tail of the distribution is not particularly long. Indeed, the maximum error distances are 5.24 km, 8.1 km, and 13.2 km for Planetlab, residential, and online maps datasets, respectively. The figure shows that the performances of the residential and online maps datasets are very similar. This is not a surprise because the online maps dataset is dominated by residential IPs. On the other hand, our system achieves clearly better results in the Planetlab scenario. We analyze this phenomenon below.

4.2.2 Landmark density

Here, we explore the number of landmarks in the proximity of targeted IPs. The larger the number of landmarks we can discover in the vicinity of a target, the larger the probability we will be able to more accurately geolocate the targeted IP. We proceed as follows. First, we count the number of landmarks in circles of radius r, which we increase from 0 to 6 km, shown in Figure 10. Then, we normalize the number of landmarks for each radius relative to the total number of landmarks seen by all three datasets that fit into the 6 km radius. Because of such normalization, the normalized number of targets for $x = 6km$ sum up to 1. Likewise, due to normalization, the value on y-axis could be considered the landmark density.

Figure 10 shows the landmark density for the three datasets as a function of the radius. The figure shows that the landmark density is largest in the Planetlab case. This is expected because one can find a number of Web-based landmarks on a University campus. This certainly increases the probability of accurately geolocating IPs in such an environment, as we demonstrated above. The figure shows that residential targets experience a lower landmark density relative to the Planetlab dataset. At the same time, the online maps dataset shows an even lower landmark density. As shown in Figure 8, our residential dataset is more biased towards urban areas. On the contrary, the online maps provide a more comprehensive and unbiased breakdown of locations. Some of them are rural areas, where the density of landmarks is naturally lower. In summary, the landmark density is certainly a factor that clearly impacts our system's geolocation accuracy. Still, additional factors such as population density, AS- and access network level properties do play a role, as we show below.

4.2.3 Global landmark density

To understand the global landmark density (more precisely, US-wide landmark density), we evenly sample 18,000 ZIP Codes over all states in US. Figure 11 shows

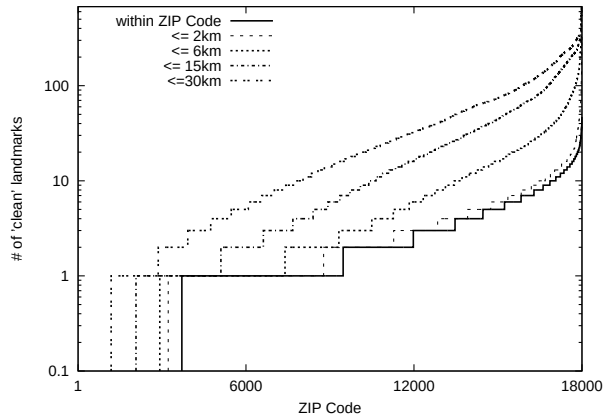

Figure 11: Number of clean landmarks for each ZIP Code

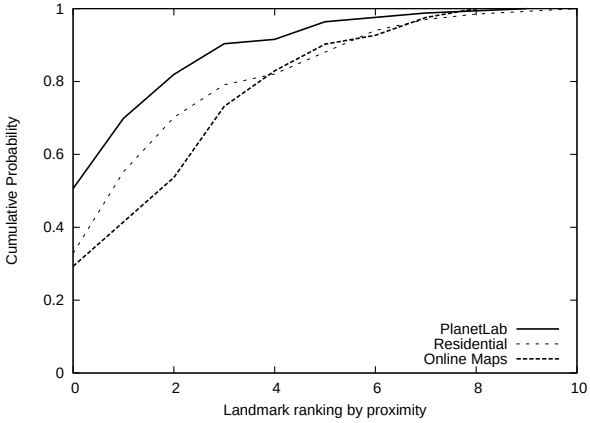

Figure 12: The distribution of the ranking of selected landmarks

that there are 79.4% ZIP Codes which contain at least one landmark within the ZIP Code. We manually check the remaining ZIP Codes and realize that they are typically the rural areas, where local entities, *e.g.*, businesses, are rare naturally. Nonetheless, for 83.78% of ZIP Codes, we are capable of finding out at least one landmark in its vicinity of 6 km; for 88.51% of ZIP Codes, we are always able to discover at least one landmark in its vicinity of 15 km; finally, for 93.44% of ZIP Codes, we find at least one landmark in its vicinity of 30 km.

We make the following comments. First, Figure 11 can be used to predict US-wide performance of our method from the *area* perspective. For example, it shows that for 6.6% of the territory, the error can only be larger than 30 km. Note, however, that such areas are extremely sparsely populated. For example, the average population density in the 6.6% of ZIP Codes that have no landmark within 30 km is less than 100. Extrapolating conservatively to the entire country, it can be computed that such areas account for about 0.92% of the entire population.

4.2.4 Relative performance

Here, we return to our datasets and evaluate our system's *relative* performance, *i.e.*, how it compares with the perfect node selection algorithm. Indeed, given that we have the information about the target's and landmarks' geographical locations, we can evaluate how close-by our approach is to the best possible node selection scenario.

Figure 12 shows the results. It depicts the cumulative probability of selecting the closest landmarks. In particular, values on the y-axis corresponding to $x = 0$ show the probability that our approach selects the best possible landmark for the three datasets. Next, values for $x = 1$ show the probability that we select one of the top two

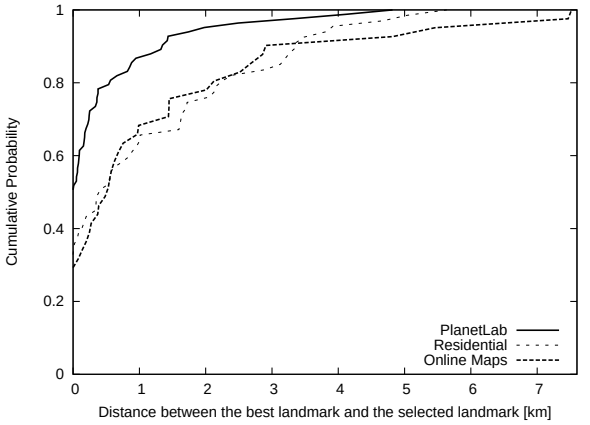

Figure 13: Performance relative to the best landmark

best landmarks, and so on. The figure demonstrates that in over 50.6%, 32.8%, and 29.3% of cases, our approach can choose the best candidate in the Planetlab, residential, and online maps datasets respectively. Moreover, the figure shows that the probability of selecting one of the top five landmarks ($x = 4$) is as high as 91.6%, 82.1% and 82.9% for the Planetlab, residential, and online maps datasets.

The reason why the Planetlab case shows better results is because to geolocate a Planetlab target, we are 30 *times* more likely to discover landmarks in the vicinity of the target that are in the same AS than is the case with the other two datasets. As we will show in Section 4.2.7, the relative delay preserves remarkably well in the same AS scenario. Thus, choosing the landmark with the minimum measured distance is more likely to hit the geographically closest landmark.

Figure 13 further evaluates how our method compares to the perfect node selection scenario. On the x-axis we

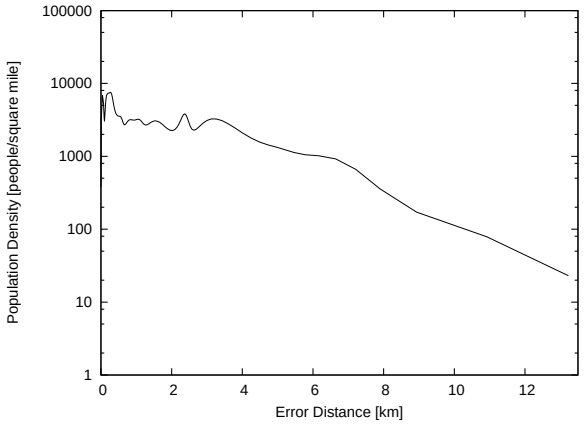

Figure 14: Error distance V.S. population density

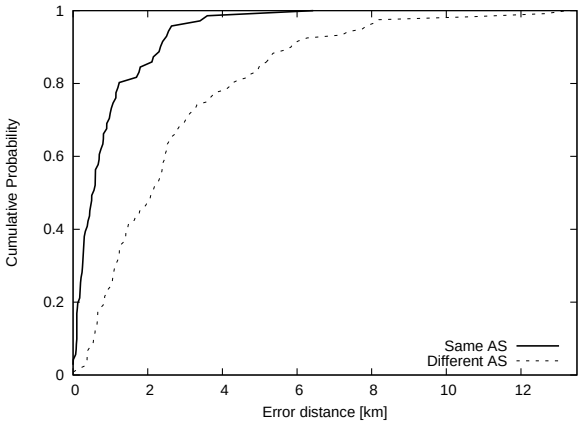

Figure 15: Comparison of error distances of same AS and different AS

show the distance between the best possible landmark candidate and our choice. This error distance can be obtained by subtracting the error distance between the target and the best candidate from that between the target and the selected candidate. Figure 13 shows that even if we fail to select the best landmark, in over 95.3%, 76.1% and 78% cases for Planetlab, residential, and online maps, respectively, the distance to the best landmark is below 2 km. Due to higher landmark density (Figure 10), and better landmark selection process (Figure 12), the performance is the best in the Planetlab case. Still, the error penalty paid by the other two approaches is not dramatic. In 95.6% cases for residential, and 91.3% for online maps, we are capable of choosing the landmark with an error distance of less than 4 km to the best landmark.

4.2.5 Tail analysis

In addition to the median error distance, the maximum error distance is another important criterion to measure geolocation accuracy. Figure 9 shows that the maximum error distance is 5.24 km in Planetlab dataset, 8.1 km in the residential dataset, and 13.2 km in the online maps dataset. We evaluate these cases in more depth. In the Planetlab's worst case, we chose the 5th best candidate with an error distance of 4.82 km to the best candidate. Thus, a close-by landmark does exist, but the system is unable to properly select it. On the contrary, the tails in the residential and online maps are dominated by the sparser landmark environments, *i.e.*, there are fewer landmarks in the vicinity of a target. We confirm this by tracing the reasons for the worst 13.2 km error that appears in the online maps scenario. We find that this is an IP in a rural area with no landmarks discovered close by, and the landmark with minimum measured distance is 13.2 km away, which we selected.

4.2.6 The role of population density

Here, we evaluate our system' s performance, *i.e.*, error distance, as a function of population density. For the sake of clarity, we merge the results of the three datasets. Figure 14 plots the best fit curve that captures the trends. It shows that the error distance is smallest in densely populated areas, while the error grows as the population density decreases. This result is in line with our analysis in Section 4.2.3. Indeed, the larger the population density is, the higher probability we can discover more landmarks. Likewise, as shown in Section 4.2.2, the more landmarks we can discover in the vicinity of targeted IP address, the higher probability we can more accurately geolocate the targeted IP. Finally, the results show that our system is still capable of geolocating IP addresses in rural areas as well. For example, we trace the IP that shows the worst error of 13.2 km. We find that this is an IP in a rural area with no landmarks discovered within the ZIP Code, which has a population density of 47.

4.2.7 The role of ASes

Here, we compare the results of the cases when the selected landmark and the target are in the same AS and in a different AS. For the sake of clarity, we merge the results of the three datasets. Figure 15 shows that the median error distance is as small as 0.56 km for the same AS case vs. 2.15 km for the different AS case. In the same AS scenario, we manage to select the closest landmarks with high accuracy, as we show below. The inter-AS case shows that while the relative distance information is necessarily more blurred, it still represents a precious geolocation resource that our method effectively utilizes.

Figure 16 shows the system performance relative to the best case scenario, in the same way as in Figure 13 above. The figure clearly indicates that landmarks from

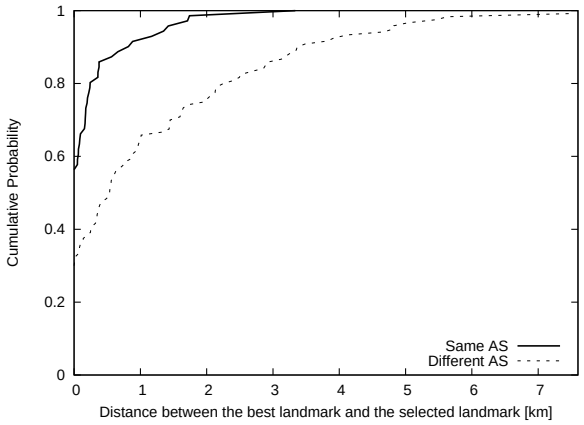

Figure 16: Performance relative to the best landmark in same AS and different AS

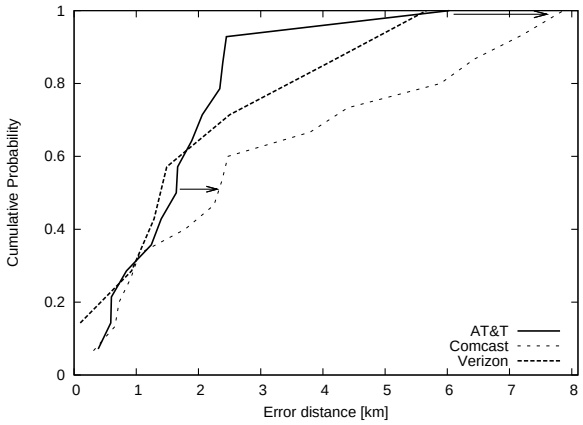

Figure 17: Comparisons of error distance in different ISPs

the same AS lead to highly accurate results. In as much as 58%, we choose the best possible landmark. Moreover, even when this is not the case, the probability of making an error larger than 1 km is smaller than 10%. The different AS scenario shows a longer tail. The probability of making an error larger than 1 km is about 35%. We explore this phenomenon in more depth below.

4.2.8 The role of access networks

Contrary to the academic environment, a number of residential IP addresses access the Internet via DSL or cable networks. Such networks create the well-known last-mile delay inflation problem, which represents a fundamental barrier to methods that rely on absolute delay measurements. Because our method relies on *relative* delay measurements, it is highly resilient to such problems, as we show below. To evaluate this issue, we examine and compare our system's performance for three different residential network providers that we collected in Section 4.1.2. These are AT&T, Comcast, and Verizon.

Figure 17 shows the CDF of the error distance for the three providers. The median error distance is 1.48 km for Verizon, 1.68 km for AT&T, and 2.38 km for Comcast. Thus, despite the fact that we measure significantly inflated delays in the last mile, we still manage to geolocate the endpoints very accurately. For example, a delay of 5 ms [14] that we commonly see at the last mile could place a scheme relying on absolute delay measurements 700 km away from the target. Our approach effectively addresses this problem and geolocates the targets within a few kilometers.

Figure 17 shows that our method has reduced performance for Comcast targets, who show a somewhat longer tail than the other two providers. We explore this issue in more depth. According to [7], AT&T and Verizon offer DSL services. Comcast is dominantly a cable Internet provider, and offers DSL only in a smaller number of areas. As demonstrated in [14], cable access networks have a much larger latency *variance*, which may rapidly vary over short time scales, than DSL networks. While our relative delay approach is resilient to *absolute* delay inflation at the last mile, it can still be hurt by measured delay variance. Because latency in cable networks changes over short time scales, it blurs our measurements, which are not fully synchronized. Hence, the landmarks' relative proximity estimation gets blurred, which causes the effects shown in the figure. In particular, the median error distance increases by approximately 700 meters relative to the DSL case (shown by the arrow from AT&T to Comcast in the middle of Figure 17), while the maximum error distance increases by 2 km (shown by the arrow from AT&T to Comcast at the top of Figure 17).

5 Discussion

Measurement overhead. Our methodology incurs measurement overhead due to Web crawling and network probing. Still, it is capable of generating near real-time responses, as we explain below. To geolocate an IP address, we crawl Web landmarks for a portion of ZIP Codes on the fly, as we explained in Sections 2.2 and 2.3. It is important to understand that this is a *one-time* overhead per ZIP Code because we cache all landmarks for every ZIP Code that we visit. Thus, when we want to geolocate other IP addresses in the vicinity of a previous one, we reuse previously cached landmarks. Once this dataset is built, only occasional updates are needed. This is because the Web-based landmarks we use are highly stable and long-lived in the common case.

On the network measurement side, we generate con-

current probes from multiple vantage points simultaneously. In the first tier, we need 2 RTTs (1 RTT from the master node to the vantage points, and 1 RTT for the ping measurements). In the second and third tiers each, the geolocation response time per IP can be theoretically limited by 3 round-trip times (1 RTT from the master node to the measurement vantage points, and 2 RTTs for an advanced traceroute overhead[2]). Thus, the total overhead on the network measurement side is 8 RTTs, which typically translates to a 1-2 seconds delay.

Migrating web services to the cloud. Cloud services are thriving in the Internet. One might have a concern that this might dramatically reduce the number of landmarks that we can rely upon. We argue that this is not the case. While more websites might indeed be served on the cloud, the total number of websites will certainly increase over time. Even if the large percent of the websites will end up in the cloud, the remaining percent of websites will always create a reliable and accurate backbone for our method. Moreover, even when an entity migrates a Web site to the cloud, the associated e-mail exchange servers do remain hosted locally (results not shown here due to space constraints). Hence, such servers can serve as accurate geolocation landmarks. Our key contribution lies in demonstrating that all such landmarks (*i.e.*, Web, e-mail, or any other) can be effectively used for accurate geolocation.

International coverage. Our evaluation is limited to US simply because we were able to obtain the vast majority of the ground-truth information from within the US. Still, we argue that our approach can be equally used in other world regions as well. This is because other countries, such as Canada, UK, other counties in EU, China, Japan, South Korea, *etc* also have their own "ZIP Code" systems. We are currently adjusting our system so that it can effectively work in these countries. Moreover, we expect that our approach will be applicable even in regions with potentially poor network connectivity. This is because our relative-delay-based method is insensitive to inflated network latencies characteristic for such environments.

[2]In the advanced traceroute case, 1 RTT is needed to obtain the IPs of intermediate routers, while another RTT is needed to simultaneously obtain round-trip time estimates to all intermediate routers by sending concurrent probes.

6 Related work

6.1 Client-independent IP geolocation systems

6.1.1 Data mining-based

DNS-based. Davis *et al.* [13] propose a DNS-based approach, which suggests adding location segments in the format of a Resource Record (RR). Nevertheless, such modification can not be easily deployed in practice and the administrators have little incentive to register or modify new RRs. Moreover, Zhang *et al.* [30] have demonstrated that DNS misnaming is common, and that it can distort Internet topology mapping.

Whois-based. Moore *et al.* [20] argue that geolocation can also be obtained by mining the Whois database. However, as the authors themselves pointed out, large entities with machines dispersed in different locations can register their domain names with the geographical location of their headquarters. As an example, many existing IP geolocation databases that use this approach incorrectly locate all Google's servers worldwide to Mountain View, CA.

Hostname-based. The machine hostnames can sometimes indicate the geolocation information. In particular, Padmanabhan's and Subramanian's GeoTrack [22] parses the location of the last access router towards the target to be located from its hostname and uses the location of this router as that of the target. For example, a router's hostname, ae-12-51.car2.Chicago2.Level3.net, indicates that this router is located in Chicago. Any target which uses this router as the last access router is also geolocated in Chicago. Unfortunately, this method can be inhibited by several factors, as pointed by [15]. First, not all machine names contain geolocation associating information. Second, administrators can be very creative in naming the machines; hence, parsing all kinds of formats becomes technically difficult. Finally, such last hop location substitution can incur errors.

Web-based. Guo *et al.*'s [18]'s Structon, mines the geolocation information from the Web. In particular, Structon builds a geolocation table and uses regular expressions to extract location information from each web page of a very large-scale crawling dataset. Since Structon does not combine delay measurement with the landmarks it discovers, it achieves a much coarser (city-level) geolocation granularity. For example, they extract all location keywords from a web page rather than just the location address. Likewise, they geolocate a domain name by choosing one from all locations provided by all the web pages within this domain name. Indeed, such approaches are error prone. Moreover, geolocating a /24 segment with a city blurs the finer-grained characteristics of each IP address in this segment. On the con-

trary, we demonstrate that information obtained from the Web is a *valuable* resource that can be effectively used to achieve fine-grained geolocation granularity. Indeed, by effectively combining information carefully extracted from the Web with network measurements, we show that significant accuracy improvements are achievable.

Other sources. Padmanabhan's and Subramanian's GeoCluster [22] geolocates IP addresses into a geographical cluster by using the address prefixes in BGP routing tables. In addition, by acquiring the geolocation information of some IP addresses in a cluster from proprietary sources, *e.g.*, users' registration records in the Hotmail service, GeoCluster deduces the location of this entire cluster. This method highly depends on the correctness of users' input and the private location information, which is in general not publicly available. Our approach differs from GeoCluster in that web designers have strong incentive to report correct location information in their websites, while users are less likely to provide accurate location information in their registration application with online services, on which GeoCluster highly relies. Moreover, we have demonstrated that using active network measurements instead of extrapolating geo information to entire clusters, is far more accurate.

6.1.2 Delay measurement-based

GeoPing. Padmanabhan and Subramanian design GeoPing [22], which assumes that two machines that have similar delay vectors tend to be close to each other. The authors rely on a set of active landmarks, *i.e.*, those capable of actively probing the target. Necessarily, the accuracy of such an approach (the comparable results shown later in the text) depends on the number of active landmarks, which is typically moderate.

CBG and its variants. Instead of yielding a discrete single geo point, Gueye *et al.* [17] introduce Constraint Based Geolocation (CBG), a method that provides a continuous geo *space* by using multilateration with distance constraints. In particular, CBG first measures the delays from all vantage points to the target. Then, it translates delays into distance by considering the best network condition of each vantage point, termed *bestline*. Finally, it returns a continuous geo space by applying multilateration. In the follow-up work, taking buffering delay [16] or bandwidth [24] into consideration, CBG's accuracy is further improved.

CBG uses bestline constraints to compensate for the fact that Internet routes are sometimes undirected or inflated. However, due to the difficulty of predicting the directness of a network route from a vantage point to a target, CBG only works well when the target is close to one of the vantage points. As explained above, we use the CBG approach straightforwardly in our tier 1 phase to discover the coarse-grained area for a targeted IP. Moreover, using newly discovered web landmarks in this area, we further constrain the targeted area in the tier 2 phase as well. Thus, while CBG is good at limiting the destination area, it is inherently limited in its ability to achieve very fine-grained resolution due to measurement inaccuracies.

TBG. Taking the advantage of the fact that routers close to the landmarks can be more accurately located, Katz-Bassett *et al.* [19] propose Topology-based Geolocation (TBG), which geolocates the target as well as the routers in the path towards the target. The key contribution of this work lies in showing that network topology can be effectively used to achieve higher geolocation accuracy. In particular, TBG uses the locations of routers in the interim as landmarks to better quantify the directness of the path to the target and geolocate it.

In addition to using network topological information, a TBG variant also takes advantage of passive landmarks with known locations. However, such an approach is constrained by the fact that it only has a very limited number of such landmarks. On the contrary, our web-based technique can conquer this difficulty significantly by discovering a large number of web-based landmarks. More substantially, TBG fundamentally relies on the *absolute* delay measurements, which are necessarily inaccurate at short distances. On the contrary, in addition to relying on a large number of web-based landmarks in an area, we demonstrate that our relative distance approach, while technically less attractive, is far more accurate.

Octant. Wong *et al.* [29] propose Octant, which considers the locations of intermediate routers as landmarks to geolocate the target. Further, Octant considers both positive information, the maximum distance that a target may be from the landmark, and negative information, the minimum distance this target may be from the landmark. In addition to delay-based constraints, Octant also enables any kind of positive and negative constraints to be deployed into its system, *e.g.*, the negative constraints (oceans and uninhabitable areas) obtained from geography and demographics.

In attempt to achieve high accuracy, Octant (as well as the above TBG method) also adopts the locations of routers in the path to the destination as landmarks to geolocate the target. However, such an approach is hampered to reach finer-grained accuracy because it fails to accurately geolocate routers at such precision in the first place. Finally, while Octant 'pushes' the accuracy of delay-based approaches to an absolutely admirable limit, it is incapable of achieving a higher precision simply due to the inherent inaccuracies associated with absolute delay measurements.

Comparative results. According to [19], TBG has

the median estimation error of 67 km that a factor of three outperforms CBG with the median estimation error of 228 km. According to [29], comparing with GeoPing and CBG, Octant with a median estimation error of 22 miles is three times better than GeoPing with an estimation error of 68 miles and four times better than CBG with an error distance of 89 miles respectively. Because TBG and Octant used the PlanetLab nodes to evaluate their system's accuracy, we can directly compare them with our system. As outlined above, our system's median error distance is 50 times smaller than Octant's, and approximately 100 times smaller than TBG's.

Network positioning systems. Our work to some extent relates to previous work that aims at designing *network* positioning systems (*e.g.*, [12, 21, 25, 28]). Such systems provide network positioning service useful for many distributed applications. These approaches measure latency among Internet hosts and landmarks, which corresponds to the network distance. Then, using virtual coordinates (*e.g.*, [12, 21, 25]) or alternative approaches (*e.g.*, [28]), they manage to determine which hosts are closer to each other in the network sense. While our work lies in the area of IP geolocation, it can have a high impact on network positioning systems as well. First, by helping better understand the network delay properties. And second, by opening avenues for more advanced *network* positioning systems.

6.2 Client-dependent IP geolocation systems

6.2.1 Wireless geolocation

GPS-based geolocation Global Positioning System (GPS) devices, that have been embedded into billions of mobile phones and computers at nowadays, could precisely provide user's location. However, GPS technology differs from our geolocation strategy in the sense that it is a 'client-side' geolocation approach, which means that the server does not know where the user is, unless the user explicitly reports his information back to the server.

Cell tower and Wi-Fi -based geolocation. Google My Location [5] and Skyhook [9] introduced their cell tower-based and Wi-Fi -based geolocation approaches. In particular, the cell tower-based geolocation offers users estimated locations by triangulating from cell towers surrounding users, while the Wi-Fi-based geolocation uses Wi-Fi access point information instead of cell towers. Specifically, every tower or Wi-Fi access point has a unique identification and footprint. To find a user's approximate location, such methods calculate user's position relative to the unique identifications and footprints of nearby cell towers or Wi-Fi access points.

Such methods could provide accurate results, *e.g.*, 200 - 1000 meters accuracy in cell tower scenario, and 10-20 meters in Wi-Fi scenario [9], on the expense of sacrificing the geolocation availability at three aspects.

First, these approaches require end user's permission to share their location. However, as we discussed above, many applications such as location-based access restrictions, context-aware security, and online advertising, can not rely on client's support for geolocation. Second, companies utilizing such an approach must deploy drivers to survey every single street and alley in tens of thousands of cities and towns worldwide, scanning for cell towers and Wi-Fi access points, as well as plotting their geographic locations. However, in our approach, we avoid such 'heavy' overhead by lightly crawling landmarks from the Web. Third, these approaches are tailored towards mobile phones and laptops. However, there are many devices (IPs) bound with wired network on the Internet. Such wireless geolocation methods are necessarily incapable of geolocating these IPs, while our method does not require any precondition on the end devices and IPs.

6.2.2 W3C geolocation

A geolocation API specification [3] is going to become a part of HTML 5 and appears to be a part of current browsers already [6].This API defines a high-level interface to location information, and is agnostic of the underlying location information sources. The underlying location database could be collected and calculated by GPS, Wi-Fi access point, cell tower, RFID, Bluetooth MAC address, as well as IP address, associated with the devices. Again, this approach requires end users' collaboration for geolocation. In addition, this method also requires browser compatibility, *e.g.*, Web browser must supports HTML 5. Finally, to geolocate wired devices, W3C geolocation has to conduct IP address-based approaches discussed in Section 6.1.1 and Section 6.1.2. In this case, our method can be considered as an effective alternative to improve the accuracy.

7 Conclusions

We have developed a client-independent geolocation system able to geolocate IP addresses with more than an order of magnitude better precision than the best previous method. Our methodology consisted of two powerful components. First, we utilized a system that effectively harvest geolocation information available on the Web to build a database of landmarks in a given ZIP Code. Second, we employed a three tiered system that begins at a large, coarse-grained, scale and progressively works its way to a finer, street-level, scale. At each stage, it takes advantage of landmark data and the fact that on

the smaller-scale, relative distances are preserved by delay measurements, overcoming many of fundamental inaccuracies encountered in the use of absolute measurements. By combining these we demonstrated the effectiveness of using both active delay measurements and web-mining for geo-location purposes.

We have shown that our algorithm functions well in the wild, and is able to locate IP addresses in the real world with extreme accuracy. Additionally, we demonstrated that our algorithm is widely applicable to IP addresses from both academic institutions, a collection of residential addresses, as well as a larger mixed collection of addresses. The high accuracy of our system in a wide range of networking environments demonstrates its potential to dramatically improve the performance of existing location-dependent Internet applications and to open the doors to novel ones.

A Proof of Probability

We see that the probability that an erroneous landmark location will cause harm to our algorithm can be computed with relative ease. Recall Figure 7, which shows us that the probability that the landmark will be safe is given by the fraction of the perimeter of the circle of radius D that lies in the light gray shaded area. First, without loss of generality, we will assume that $R = 1$. We also see that we need only consider the top half of the circle, since the diagram is symmetric about the x-axis. Next, we note that if a represents the arc of the circle that lies within the unshaded circle, the fraction that lies in the safe shaded region is given by

$$1 - \frac{a}{\pi}.$$

So it remains only to determine the arc a for a circle of radius D. We observe that the centers of the two circles, and the point of intersection form a triangle, and it is clear that the angle θ at the center of the circle of radius D gives a. Direct application of the law of cosines reveals that $a = cos^{-1}D/2$ and therefore our probability is given by:

$$1 - \frac{cos^{-1}(D/2)}{\pi}$$

References

[1] City data. http://www.city-data.com/.

[2] Geolocation and application delivery. www.f5.com/pdf/white-papers/geolocation-wp.pdf.

[3] Geolocation api specification. http://dev.w3.org/geo/api/spec-source.html.

[4] Geonames. http://www.geonames.org/.

[5] Google maps with my location. http://www.google.com/mobile/gmm/mylocation/index.html.

[6] How google maps uses the w3c geolocation api and google location services. http://apb.directionsmag.com/archives/6094-How-Google-Maps-uses-the-W3C-Geolocation-API-and-Goo html.

[7] Ooakla's speedtest throughput measures. http://silicondetector.org/display/IEPM/Ookla%27s+Speedtest+Throughput+Measures.

[8] Reverse ip domain check. http://www.yougetsignal.com/tools/web-sites-on-web-server/.

[9] Skyhook. http://www.skyhookwireless.com/.

[10] CHENG, H., ANGELA, W., JIN, L., AND W, R. K. Measuring and evaluating large-scale CDNs. In *Microsoft Technical Report*.

[11] CLARK, D., PARTRIDGE, C., BRADEN, R., DAVIE, B., FLOYD, S., JACOBSON, V., KATABI, D., MINSHALL, G., RAMAKRISHNAN, K., ROSCOE, T., STOICA, I., WROCLAWSKI, J., AND ZHANG, L. Making the world (of communications) a different place. *ACM Computer Comm. Review* (2005).

[12] DABEK, F., COX, R., KAASHOEK, F., AND MORRIS, R. Vivaldi: A decentralized network coordinate system. In *SIGCOMM, '04*.

[13] DAVIS, C., VIXIE, P., GOODWIN, T., AND DICKINSON, I. A means for expressing location information in the domain name system. *RFC 1876* (1996).

[14] DISCHINGER, M., HAEBERLEN, A., GUMMADI, K. P., AND SAROIU, S. Characterizing residential broadband networks. In *IMC, '07*.

[15] FREEDMAN, M. J., VUTUKURU, M., FEAMSTER, N., AND BALAKRISHNAN, H. Geographic locality of ip prefixes. In *IMC, '05*.

[16] GUEYE, B., UHLIG, S., ZIVIANI, A., AND FDIDA, S. Leveraging buffering delay estimation for geolocation of internet hosts. In *IFIP Networking Conference, '06*.

[17] GUEYE, B., ZIVIANI, A., CROVELLA, M., AND FDIDA, S. Constraint-based geolocation of internet hosts. *Transactions on Networking* (2006).

[18] GUO, C., LIU, Y., SHEN, W., WANG, H. J., YU, Q., AND ZHANG, Y. Mining the web and the internet for accurate ip address geolocations. In *Infocom mini conference, '09*.

[19] KATZBASSETT, E., JOHN, J. P., KRISHNAMURTHY, A., WETHERALL, D., ANDERSON, T., AND YATIN. Towards ip geolocation using delay and topology measurements. In *IMC, '06*.

[20] MOORE, D., PERIAKARUPPAN, R., DONOHOE, J., AND CLAFFY, K. Where in the world is netgeo.caida.org? In *INET '00*.

[21] NG, T. S. E., AND ZHANG, H. Predicting internet network distance with coordinates-based approaches. In *INFOCOM, '02*.

[22] PADMANABHAN, V. N., AND SUBRAMANIAN, L. An investigation of geographic mapping techniques for internet host. In *ACM SIGCOMM '01*.

[23] PERCACCI, R., AND VESPIGNANI, A. Scale-free behavior of the internet global performance. *The European Physical Journal B - Condensed Matter* (2003).

[24] SIWPERSAD, S., BAMBAGUEYE, AND UHLIG, S. Assessing the geographic resolution of exhaustive tabulation for geolocating internet hosts. In *PAM, '08*.

[25] TANG, L., AND CROVELLA, M. Virtual landmarks for the internet. In *IMC, '03*.

[26] VALANCIUS, V., LAOUTARIS, N., MASSOULIE, L., DIOT, C., AND RODRIGUEZ, P. Greening the internet with nano data centers. In *CONEXT 09*.

[27] VINCENTY, T. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review* (1975).

[28] WONG, B., SLIVKINS, A., AND SIRER, E. G. Meridian: A lightweight network location service without virtual coordinates. In *ACM SIGCOMM, '05*.

[29] WONG, B., STOYANOV, I., AND SIRER, E. G. Octant: A comprehensive framework for the geolocalization of internet hosts. In *NSDI, '07*.

[30] ZHANG, M., RUAN, Y., PAI, V., AND REXFORD, J. How dns misnaming distorts internet topology mapping. In *USENIX Annual Technical Conference, '06*.

[31] ZIVIANI, A., FDIDA, S., DE REZENDE, J. F., AND DUARTE, O. C. M. Improving the accuracy of measurement-based geographic location of internet hosts. *Computer Networks, Elsevier Science* (2005).