How to Catch when Proxies Lie

Verifying the Physical Locations of Network Proxies with Active Geolocation

Zachary Weinberg
Carnegie Mellon University
zackw@cmu.edu

Shinyoung Cho
Stony Brook University
shicho@cs.stonybrook.edu

Nicolas Christin
Carnegie Mellon University
nicolasc@cmu.edu

Vyas Sekar
Carnegie Mellon University
vsekar@cmu.edu

Phillipa Gill
University of Massachusetts
phillipa@cs.umass.edu

ABSTRACT

Internet users worldwide rely on commercial network proxies both to conceal their true location and identity, and to control their apparent location. Their reasons range from mundane to security-critical. Proxy operators offer no proof that their advertised server locations are accurate. IP-to-location databases tend to agree with the advertised locations, but there have been many reports of serious errors in such databases.

In this study we estimate the locations of 2269 proxy servers from ping-time measurements to hosts in known locations, combined with AS and network information. These servers are operated by seven proxy services, and, according to the operators, spread over 222 countries and territories. Our measurements show that one-third of them are definitely not located in the advertised countries, and another third might not be. Instead, they are concentrated in countries where server hosting is cheap and reliable (e.g. Czech Republic, Germany, Netherlands, UK, USA).

In the process, we address a number of technical challenges with applying active geolocation to proxy servers, which may not be directly pingable, and may restrict the types of packets that can be sent through them, e.g. forbidding `traceroute`. We also test three geolocation algorithms from previous literature, plus two variations of our own design, at the scale of the whole world.

CCS CONCEPTS

• **Networks** → **Network measurement**; *Network structure*;

KEYWORDS

active geolocation, virtual private networks, network proxies

ACM Reference Format:
Zachary Weinberg, Shinyoung Cho, Nicolas Christin, Vyas Sekar, and Phillipa Gill. 2018. How to Catch when Proxies Lie: Verifying the Physical Locations of Network Proxies with Active Geolocation. In *2018 Internet Measurement Conference (IMC '18), October 31–November 2, 2018, Boston, MA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3278532.3278551

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IMC '18, October 31–November 2, 2018, Boston, MA, USA
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-5619-0/18/10…$15.00
https://doi.org/10.1145/3278532.3278551

1 INTRODUCTION

Commercial VPN services compete to offer the highest speed, the strongest privacy assurances, and the broadest possible set of server locations. One of the services in this study advertises servers in all but seven of the world's sovereign states, including implausible locations such as North Korea, Vatican City, and Pitcairn Island. They offer no proof of their claims. IP-to-location databases often agree with their claims, but these databases have been shown to be full of errors [18, 38, 42]. Worse, they rely on information that VPN providers may be able to manipulate, such as location codes in the names of routers [7].

VPN services that consolidate their servers in a smaller number of locations than they advertise can choose those locations for better performance, reliability, and reduced operational expenses. This gives them a competitive advantage over services that strive for true location diversity. If they can manipulate IP-to-location databases, they can still provide the *appearance* of location diversity.

Many of a VPN service's customers may well be satisfied with appearances. For instance, the IP-to-location database entry is more important than the physical location for customers using VPNs to defeat geographic restrictions on online media streaming [2]. However, for others the physical location can be essential. We started the investigation leading to this paper when we attempted to use commercial VPN services for censorship monitoring, but could not reproduce the observations reported by volunteers within a country known for censoring the Internet.

In this paper, we apply *active geolocation* to check the advertised locations of VPN servers. Active geolocation estimates the location of an Internet host by measuring packet round-trip times between it and other hosts in known locations. It has been demonstrated to work at the scale of a large country or small continent (e.g. China, Europe, and the USA), with varying levels of accuracy, depending on how efficient the regional network is [8, 11, 16, 32]. However, it has not been thoroughly tested at the scale of the entire world, and, to our knowledge, it has only once before been applied to commercial proxy servers [39].

Using active geolocation, we can usually locate a VPN server to within 1000 km^2, anywhere in the world. Our results are more precise in more densely connected regions and/or when landmarks are nearby, but even when we are uncertain about where a server actually is, we can still disprove blatant inaccuracies in marketing claims. For instance, if we know that a server is in Belgium, Netherlands, or Germany, but not which, that still proves it is not in North Korea. We tested 2269 servers operated by seven VPN services,

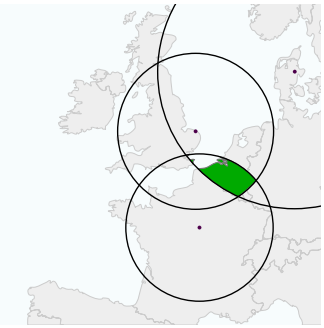

Figure 1: The principle of multilateration. If something is within 500 km of Bourges, 500 km of Cromer, and 800 km of Randers, then it is in Belgium (roughly).

including five of the top 20 by number of claimed countries. *At least a third of all the servers we tested are not in their advertised country.*

2 BACKGROUND

Existing methods for finding the physical location Internet hosts can be divided into two general classes. Passive methods collect location information from regional Internet registries, location information encoded in router hostnames, and private consultation with individual ISPs [7], and produce a database mapping IP addresses to locations. These databases are widely used, but notorious for their errors [18, 38, 42], some of which are significant enough that they make the news [21].

Active methods, on the other hand, rely on measurements of packet round-trip time between a *target* host, which is to be located, and a number of *landmark* hosts, which are in known locations. The simplest active method is to guess that the target is in the same place as the landmark with the shortest round-trip time [8, 35, 46]. This breaks down when the target is not near any of the landmarks. The next step up in complexity is to estimate, for each landmark, the maximum distance that a packet could have traveled in the time measured, and draw disks on a map, bounded by these distances. The target must be in the region where the disks all intersect. This process is called *multilateration*. Figure 1 shows an example: measurements taken from Bourges in France, Cromer in the UK, and Randers in Denmark produce an intersection region roughly covering Belgium.

The central problem for network multilateration is that network packets do not travel in straight lines. Cables are laid on practical paths, not great circles. Network routes are optimized for bandwidth rather than latency, leading to "circuitous" detours that can add thousands of kilometers to the distance traveled [29, 31, 34]. Intermediate routers can add unbounded delays [32]. Distance and delay do still correlate, but not in a straightforward way. Much research on active methods focuses on increasingly sophisticated models of the delay-distance relationship [4, 12, 14, 20, 30, 31, 34, 35, 45]. One common refinement is to assume a minimum travel distance for any given delay, as well as a maximum.

Challenges of global geolocation. When both landmarks and targets are in the same subcontinental region, sophisticated models improve accuracy—if that region is Europe or the USA. On the other hand, for China, several papers report that *simple* models are more accurate [8, 11, 32]. They propose that simple models are more robust in the face of severe congestion. Li et al. [32] specifically points out that a minimum travel distance assumption is invalid in the face of large queueing delays at intermediate routers. A second possibility is that sophisticated models are more reliable when there are more possible paths between landmarks and targets, as is the case in Europe and North America, but not China [16]. A third is that models tested on PlanetLab nodes [37] gain an unfair advantage due to the generally better connectivity enjoyed by academic networks. In Section 5, we test four algorithms, covering a range of model complexity, on hosts crowdsourced from all over the world. We also find that simple models are more effective, overall, and our data is more consistent with the congestion explanation.

Increasing the number of landmarks improves accuracy but also slows down the measurement process, since all of the landmarks must send packets to the target and wait for replies (or vice versa). If they all do this simultaneously, they may create enough extra network congestion to invalidate the measurement [22]. Several researchers have observed that landmarks far away from the target are less useful, and proposed a two-stage process, in which a small number of widely dispersed landmarks identify the subcontinental region where the target lies, and then a larger group of landmarks within that region pin down the target's position more accurately [11, 23, 26, 46].

Challenges of geolocating proxies. Less than ten percent of the proxies we are interested in testing will respond to pings, and we do not have the ability to run measurement programs on the proxies themselves. We can only send packets *through* the proxies, which means the apparent round-trip time to each landmark is the sum of the round-trip time from the proxy to the landmark, and the round-trip time from our measurement client to the proxy. This is similar to the problem faced by Castelluccia et al. [5] when attempting to geolocate botnet command-and-control servers, and we adopt the same solution, as discussed further in Section 5.3.

3 ALGORITHM SELECTION

Since the proxies we are investigating could be spread all over the world, we must find an active geolocation algorithm that will work at the scale of the whole world. We reimplemented four active geolocation algorithms from earlier papers: CBG [20], Octant [45], Spotter [30], and an Octant/Spotter hybrid of our own invention. We did not have access to the original implementations, and we had to fill in gaps in all their published descriptions. All the software we developed for this project is open-source and available online.[1]

Eriksson et al. [15] recommend considering external facts about where a server could plausibly be, such as "on land, and not in Antarctica." We take this advice and exclude all terrain north of 85° N and south of 60° S from the final prediction region for each target and algorithm. Using the 2012 Natural Earth [36] map of the world, we also exclude oceans and lakes. We do not, however, exclude any islands, no matter how small or remote, because some of the proxy providers do claim to have servers on remote islands (e.g. Pitcairn).

[1] https://github.com/zackw/active-geolocator

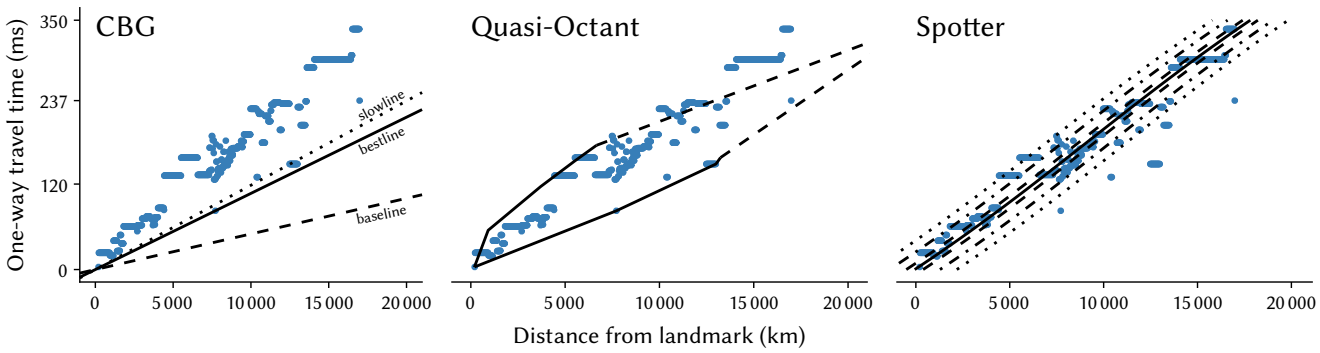

Figure 2: Example calibration scatterplots for CBG, (Quasi-)Octant, and Spotter.

3.1 Constraint-Based Geolocation

Constraint-Based Geolocation (CBG) is one of the oldest and simplest multilateration algorithms. It uses a linear model for the delay-distance relationship, limited by a "baseline" speed of 200 km/ms, or $\frac{2}{3}c$, which is approximately how fast signals propagate in fiber-optic cable. For each landmark, CBG computes a "bestline" from the calibration data, which is as close as possible to all of the data points on a scatterplot of delay as a function of distance, while remaining below all of them, and above the baseline. This will be a speed *slower* than 200 km/ms, and will therefore give a *smaller* estimate of how far a packet could have gone in a given time. Each landmark's bestline gives the maximum distance for a round-trip measurement to that landmark.

The left panel of Figure 2 shows an example calibration for CBG. The blue dots are round-trip time measurements taken by one RIPE anchor. The bestline (solid) is above the baseline (dotted); it passes through exactly two data points, with all the others above. It corresponds to a speed of 93.5 km/ms—less than half the theoretical maximum. The "slowline" will be explained in Section 5.1.

3.2 Quasi-Octant

Octant elaborates on CBG in two ways. First, it estimates both the maximum and the minimum distance to each landmark, and draws rings on the map, not disks. Second, Octant uses piecewise-linear curves for both distance models. These are defined by the convex hull of the scatterplot of delay as a function of distance, up to 50% and 75% of all round-trip times, respectively. Observations beyond those cutoffs are considered unreliable, so Octant uses fixed empirical speed estimates for longer round-trip times. The middle panel of Figure 2 shows an example Octant calibration, with the same data as the CBG calibration to its left. The convex hull is drawn with solid lines and the fixed empirical speeds with dashed lines.

Octant includes features that depend on route traces, such as a "height" factor to eliminate the effect of a slow first hop from any given landmark. Since we cannot collect route traces (see Section 4.2), these have been omitted from our re-implementation, and we call it "Quasi-Octant" to denote that change.

3.3 Spotter

Spotter [30] uses an even more elaborate delay-distance model. It computes the mean and standard deviation of landmark-landmark distance as a function of delay, and fits "a polynomial" to both. Unlike CBG and Octant, a single fit is used for all landmarks. The paper does not specify the degree of the polynomial, or the curve-fitting procedure; we use cubic polynomials, fit by least squares, and constrain each curve to be increasing everywhere (anything more flexible led to severe overfitting in pilot tests).

Spotter also uses a probabilistic multilateration method. It estimates the distance from each landmark to the target as a Gaussian distribution, with mean μ and standard deviation σ given by the fitted curves. This produces a ring-shaped probability distribution over the surface of the Earth; the rings for each landmark are combined using Bayes' Rule to form the final prediction region.

The right panel of Figure 2 shows an example Spotter calibration. The solid line is the best cubic fit for the mean μ of the distance-delay relationship; dashed, dash-dot, and dotted lines are drawn at $\mu \pm \sigma$, $\mu \pm 3\sigma$, and $\mu \pm 5\sigma$ respectively.

3.4 Quasi-Octant/Spotter Hybrid

To separate the effect of Spotter's probabilistic multilateration from the effect of its cubic-polynomial delay model, we also implemented a hybrid that uses Spotter's delay model, but Quasi-Octant's ring-based multilateration. The minimum and maximum radii of the ring are set to $\mu - 5\sigma$ and $\mu + 5\sigma$, respectively.

4 MEASUREMENT METHOD

For all our experiments, we used the "anchor" hosts of RIPE Atlas [40] as landmarks. RIPE Atlas is a worldwide constellation of hosts dedicated to Internet measurement, composed of "probes" and "anchors;" there are fewer anchors, but they are more convenient for use as landmarks. They are reliably available 24/7, their documented locations are accurate, and they all continuously ping each other and upload the round-trip times (RTT) to a publicly accessible database. At the time we began our experiments (July 2016), there were 207 usable anchors; during the course of the experiment, 12 were decommissioned and another 61 were added. Figure 3 (left side) shows all the anchors' locations. The majority

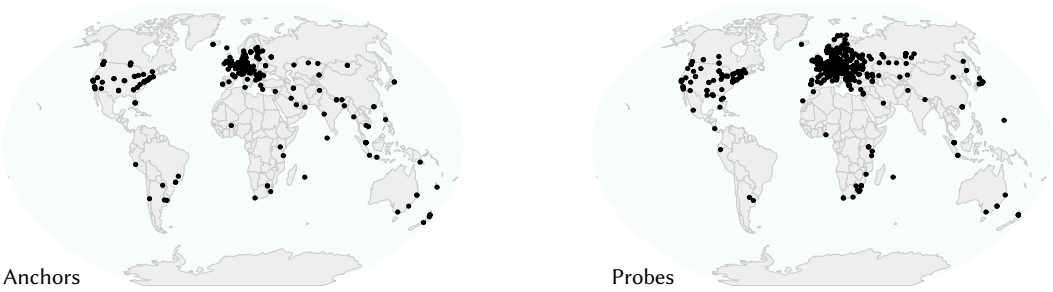

Figure 3: Locations of the RIPE Atlas anchors (left) and probes with stable IPv4 addresses, as of April 2018.

are in Europe; North America is also well-represented. While there are fewer anchors in Asia and South America, and only a few in Africa, their geographic distribution is adequate—the most difficult case for active geolocation is when all of the landmarks are far away from the target, in the same direction [16, 34].

4.1 Two-phase measurement

It takes several minutes to ping all 250 of the anchors. Landmarks far from the target do not contribute much useful information, as we will discuss further in Section 5.2. We speed up the process with a two-phase measurement, as proposed by Khan et al. [26] and others [11, 23, 46]. We first measure RTTs to three anchors per continent, and use these measurements to deduce which continent the target is on. We then randomly select and measure RTTs to 25 more landmarks on that continent, from a list including all of the anchors, plus all the probes that have been online for the past 30 days with a stable IPv4 address. These probes are shown in Figure 3 (right side).

Random selection of landmarks in the second phase spreads out the load of our measurements, reducing their impact on concurrent experiments [22]. Using stable probes as well as anchors spreads the load even in parts of the world where there are few anchors.

We maintain a server that retrieves the list of anchors and probes from RIPE's database every day, selects the probes to be used as landmarks, and updates a delay-distance model for each landmark, based on the most recent two weeks of ping measurements available from RIPE's database. Our measurement tools retrieve the set of landmarks to use for each phase from this server, and report their measurements back to it. Some of the landmarks have both IPv4 and IPv6 addresses, but the commercial proxy servers we are studying offer only IPv4 connectivity, so the server resolves the landmarks' hostnames itself and sends only IPv4 addresses to the tools.

4.2 Measurement tools

Commercial proxy providers aggressively filter traffic through their proxies. Of the VPN servers we tested, roughly 90% ignore ICMP ping requests. Similarly, 90% of the default gateways for VPN tunnels (i.e. the first-hop routers for the VPN servers) ignore ping requests and do not send time-exceeded packets, which means we cannot see them in a `traceroute` either. Roughly a third of the servers discard *all* time-exceeded packets, so it is not possible to `traceroute` through them at all. Some servers even drop UDP and TCP packets with unusual destination port numbers.

In short, the only type of network message we can reliably use to measure round-trip time is a TCP connection on a commonly used port, e.g. 80 (HTTP). We implemented two measurement tools that use this method to measure round-trip times to each landmark.

Command-line. For measurements of VPN proxies' locations (Section 6), we used a standalone program, written in Python and C. It can take measurements either directly or through a proxy, and it can process a list of proxies in one batch.

This tool uses the POSIX sockets API to make TCP connections. It measures the time for the `connect` primitive to succeed or report "connection refused," and then closes the connection without sending or receiving any data. We verified that `connect` consistently returns as soon as the second packet of the TCP three-way handshake arrives (i.e. after a single round-trip to a landmark) on both Linux and NetBSD. (Linux was used as the client OS for all the measurements of VPN proxies; some pilot tests involved clients running NetBSD.) If a connection fails with an error code other than "connection refused," the measurement is discarded. "Network unreachable" errors, for instance, originate from intermediate routers, so they do not reflect the full round-trip time.

Web-based. For algorithm validation (Section 5) we crowdsourced hosts in known locations from around the world. We could not expect volunteers from the general public, or Mechanical Turk workers, to download, compile, and run a command-line tool, so we implemented a second measurement tool as a Web application. Anyone can access the website hosting this application,[2] and it requires no "plug-ins" or software installation. It presents a live demonstration of active geolocation, displaying the measurements as circles drawn on a map, much as in Figure 1. After this demonstration is complete, it offers an explanation of the process, and invites the user to upload the measurements to our database, if they are willing to report their physical location.

The price of user-friendliness is technical limitations. Web applications are only allowed to send well-formed HTTP(S) messages; we cannot close connections immediately upon establishment, without sending or receiving any data, as the command-line tool does.

In principle, web applications are not allowed to communicate with arbitrary hosts, only with the server hosting their website [24]. However, this rule has a loophole. When a web application attempts to communicate with a server that isn't hosting its website, the browser will send an HTTP request, but won't return a successful

[2] https://research.owlfolio.org/active-geo

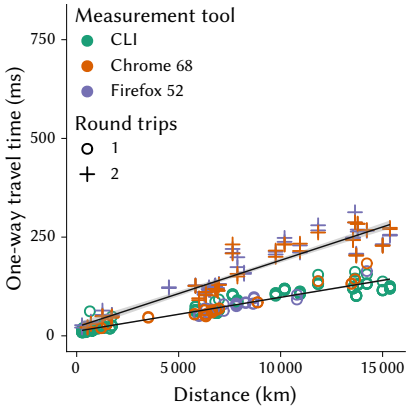

Figure 4: Comparison of the CLI tool with the web application in two browsers, all running on Linux.

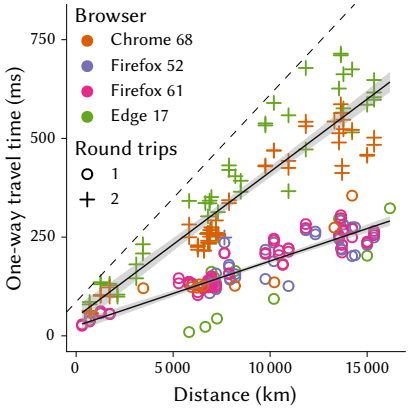

Figure 5: Comparison of four browsers running on Windows 10.

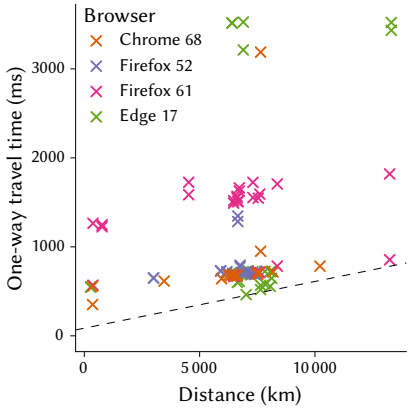

Figure 6: High outliers removed from Figure 5. The dashed line has the same slope in both figures.

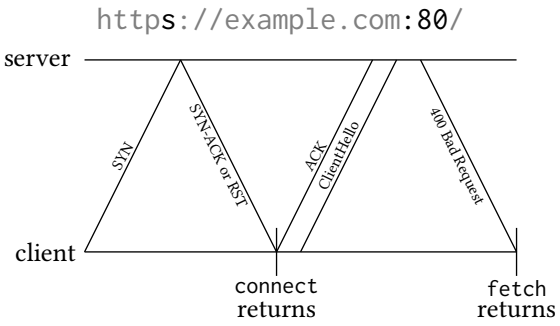

Figure 7: Both measurement tools make TCP connections to port 80 on each landmark. The CLI tool can use the low-level `connect` API; the web application must use the higher-level `fetch` API. We instruct `fetch` to send encrypted (HTTPS) traffic to the usual port for *unencrypted* HTTP, forcing a protocol error. The CLI tool reliably measures one round-trip time; the web application measures one or two round-trips, depending on whether the landmark is listening on port 80.

response unless the server allows it, using special HTTP response headers. Errors are still reported. Since we only care about the time to connect, we make a request that we know will fail, and measure the time to failure. Ideally, we would connect to a TCP port that was not blacklisted by any VPN provider, and was closed (not blackholed) on all the RIPE Atlas nodes we use, but there is no such port.

Instead, the web application makes encrypted (HTTPS) connections to the usual TCP port for *unencrypted* HTTP (80). This will fail after one round-trip if the landmark isn't listening on port 80. However, if it is listening on port 80, the browser will reply to the SYN-ACK with a TLS ClientHello packet. This will trigger a protocol error, and the browser will report failure, but only after a second round-trip. Thus, depending on whether the landmark is listening on port 80 (which depends on the version of the RIPE Atlas node software it is running; we cannot tell in advance) the web application will measure the time for either one or two round-trips, and we can't tell which.

4.3 Tool Validation

Figure 7 shows the abstract difference in the network traffic generated by the two tools. Figure 4 compares the round-trip times measured by the command-line tool and the web application running under two different browsers, all three on a computer in a known location running Linux, to a collection of landmarks as described in Section 4.1. We manually partitioned the measurements taken by the web application into groups suspected to be one round trip and two round trips, and estimated the distance-delay relationship for each by linear regression, shown with black lines and gray 95% confidence intervals. The slope of the two-round-trip line is 1.96 times the slope of the one-round-trip line; adjusted R^2 (considering both lines as a single model) is 0.9942. After accounting for the effects of distance and whether we measured one or two round trips, ANOVA finds no significant difference among the three tools (two additional degrees of freedom, $F = 0.8262$, $p = 0.44$) which is a testament to the efficiency of modern JavaScript interpreters.

Figure 5 compares the round-trip times measured by the web application running under four additional browsers, on the same computer that was used for Figure 4, but running Windows 10 instead. (The command-line tool has not been ported to Windows.) Measurements on Windows are much noisier than on Linux. We can still distinguish a group of one-round-trip data points and a group of two-round-trip data points, but there is a third group, "high outliers," separately shown in Figure 6 so that Figures 4 and 5 can have the same vertical scale. The diagonal dashed line on Figures 5 and 6 has the same absolute slope. The high outlier measurements are much slower than can be attributed to even two round-trips, and their values are primarily dependent on the browser they were measured with, rather than the distance.

Excluding the high outliers, the remaining data points for Windows can also be modeled by a division into groups for one or two round-trips, but not as cleanly as on Linux. The ratio of slopes is

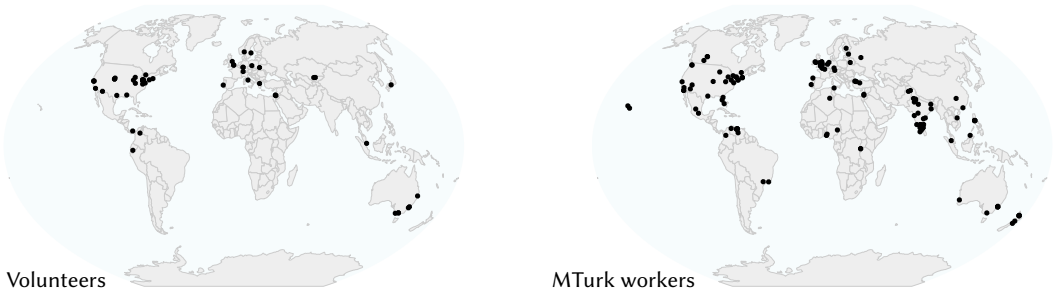

Figure 8: Locations of the crowdsourced hosts used for algorithm validation, with volunteers on the left and Mechanical Turk workers on the right.

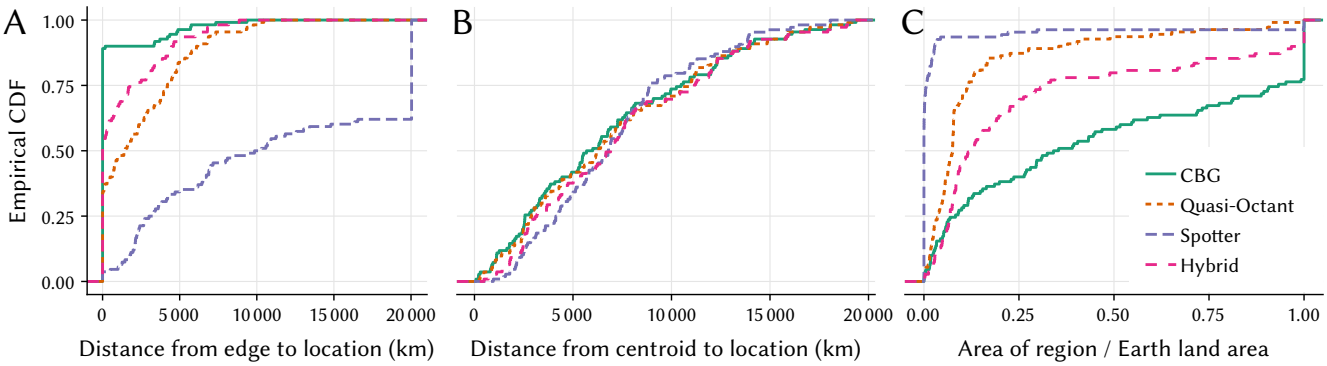

Figure 9: Precision of predicted regions for crowdsourced test hosts.

2.29, adjusted $R^2 = 0.8983$, and ANOVA finds the model is significantly improved by considering the browser as well (three more degrees of freedom, $F = 13.11$, $p = 6.1 \times 10^{-8}$). Equally concerning, if we combine the two models, we find that the operating system has a significant effect on the slopes of the lines (four additional degrees of freedom, $F = 693.56$, $p < 2.2 \times 10^{-16}$) and the regression line for two round-trips measured on Linux ($t = 0.03375d + 45.52$, distance in km, time in ms) is about the same as the line for *one* round-trip measured on Windows ($t = 0.03288d + 49.92$).

In section 5, we will speak further of how these limitations affect our assessment of which algorithm is most suitable for estimating the location of a proxy that could be anywhere in the world.

5 ALGORITHM TESTING

In order to test our geolocation algorithms on hosts they hadn't been calibrated with, we crowdsourced a second set of hosts in known locations.[3] 40 volunteers, recruited from a variety of mailing lists and online forums, and another 150 paid contributors, recruited via Mechanical Turk for 25¢ each, provided us with the approximate physical location of their computers (rounded to two decimal places in latitude and longitude, or roughly 10 km of position uncertainty) and a set of round-trip times to RIPE Atlas anchors and probes, using the Web-based measurement tool described in Section 4.2. Their self-reported locations are shown in Figure 8. Like the RIPE anchors, the majority are in Europe and North America, but we have enough contributors elsewhere for statistics.

Our priority was to find an algorithm that would always include the true location of each host in its predicted region, even if this meant the region was fairly imprecise. To put it another way, when investigating the locations of commercial proxies, we want to be certain that the proxy is where we say it is, even if that means we cannot assure that it is not where the provider says it is.

In figure 9, panel A, we plot an empirical CDF of how far outside the predicted region each true location is, for each of the four algorithms. This is a direct measure of each algorithm's failure to live up to the above requirement. None of the algorithms are perfect, but CBG does better than the other three, producing predictions that do include the true location for 90% of the test hosts, and are off by less than 5000 km for 97% of them. Hybrid and Quasi-Octant's predictions miss the mark for roughly 50% of the test hosts, but they are off by less than 5000 km for roughly 90%. Fully half of Spotter's predictions are off by more than 10 000 km.

In panels B and C of Figure 9, we look into *why* the predictions miss the true region. Panel B shows that the distances from the *centroid* of each algorithm's predictions, to the true locations, are about the same for all four algorithms, and panel C shows that CBG produces predictions that are much larger than the other three. We conclude that none of the algorithms can reliably center their predicted region on the true location, but CBG's predictions are

[3]This study was approved by our university's IRB.

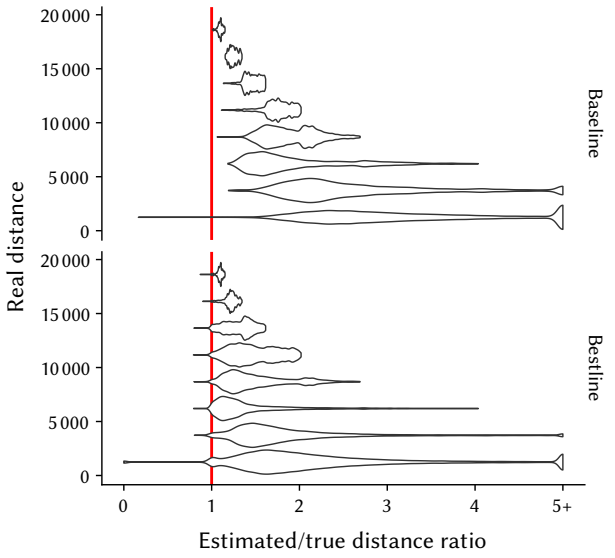

Figure 10: CBG bestline and baseline estimates compared to the true distance.

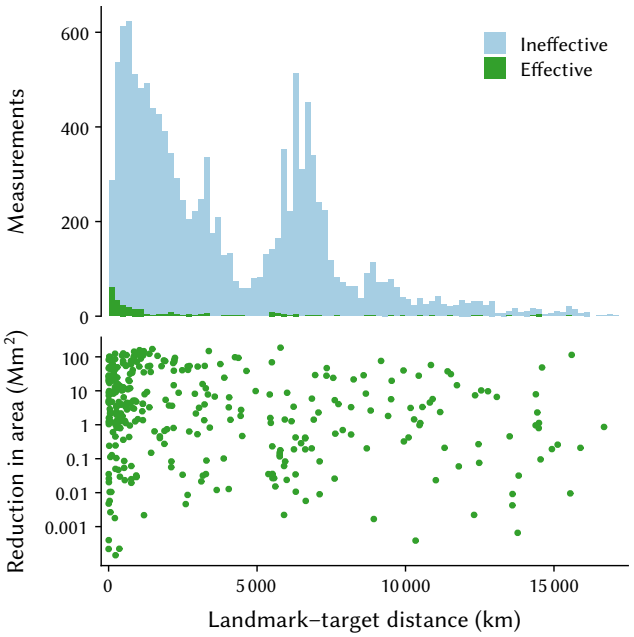

Figure 11: Proportion of measurements that had an effect on the final prediction region, as a function of distance between landmark and target; for effective measurements, the amount by which they reduced the size of the final region. The total land area of Earth is roughly 150 square megameters (Mm^2), and the land area of Egypt is roughly 1 Mm^2.

usually big enough to cover the true location anyway, whereas the other three algorithms' predictions are not big enough.

Why should CBG be so much more effective? Looking again at the calibration data in Figure 2, we observe that most of the data points are well above CBG's bestline. Quasi-Octant and Spotter draw more information from these points than CBG does. If most of those points are dominated by queueing and circuitousness delays, rather than the great-circle distance between pairs of landmarks, that would lead Quasi-Octant and especially Spotter to underestimate the speed packets can travel, therefore predicting regions that are too small. Large queueing delays also invalidate the assumption, shared by both Quasi-Octant and Spotter, that there is a *minimum* speed packets can travel [32].

Most of our crowdsourced contributors used the web application under Windows. As we described in Section 4.3, this introduces extra noise and "high outliers" into the measurements. CBG has an inherent advantage in dealing with measurements biased upward, since it always discards all but the quickest observation for each landmark, its bestlines are the fastest travel time consistent with the data, and it does not assume any minimum travel speed when multilaterating. Crowdsourced measurements using only the command-line tool might have allowed Quasi-Octant and Spotter to do better. However, measurements taken through proxies are liable to suffer extra noise and queuing delays as well. We could thus argue that the web application's limitations make the crowdsourced test a better simulation of the challenges faced by active geolocation of proxies.

5.1 Eliminating Underestimation: CBG++

Regardless of the reasons, CBG clearly is the most effective algorithm in our testing, but it still doesn't always cover the true location with its predictions. We made two modifications in order to eliminate this flaw, producing a new algorithm we call CBG++.

CBG's disks can only fail to cover the true location of the target if some of them are too small. A disk being too small means the corresponding bestline underestimates the distance that packets could travel. This can easily happen, for instance, when the network near a landmark was congested during calibration [28]. Not only can an underestimate make the prediction miss the target, it can make the intersection of all the disks be empty, meaning that the algorithm fails to predict *any* location for the target.

To reduce the incidence of underestimation, we first introduced another physical plausibility constraint. CBG's bestlines are constrained to make travel-speed estimates no faster than 200 km/ms as packets can travel no faster than this in undersea cables. We also constrain them to make travel-speed estimates no *slower* than 84.5 km/ms; this is the "slowline" in the CBG panel of Figure 2. The logic behind this number is: No landmark can be farther than half the equatorial circumference of the Earth, 20 037.508 km, from the target. One-way travel times greater than 237 ms could have involved a geostationary communications satellite, and one such hop can bridge any two points on the same hemisphere, so they provide no useful information. 20 037.508 km/237 ms = 84.5 km/ms.

The slowline constraint is not enough by itself. Figure 10 shows the distribution of ratios of bestline and baseline distance estimates to the true distances, for all pairs of landmarks, with the slowline

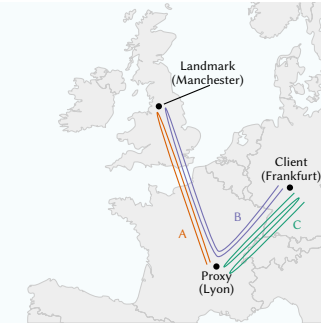

Figure 12: The RTT from a proxy to a landmark, A, must be derived from the RTT through a proxy to a landmark, B, and the RTT through a proxy back to the client, C: $A = B - \eta C$.

constraint applied. We use the landmarks themselves for this analysis, rather than the crowdsourced test hosts, because we know their positions more precisely and the ping-time measurements they make themselves are also more accurate. A small fraction of all bestline estimates are still too short, and for very short distances this can happen for baseline estimates as well.

We weed out the remaining underestimates with a more sophisticated multilateration process. For each landmark, we compute both the bestline disk, and a larger disk using the baseline. We find the largest subset of all the baseline disks whose intersection is nonempty; this is called the "baseline region." Any bestline disk that does not overlap this region is discarded. Finally we find the largest subset of the remaining bestline disks whose intersection is nonempty; this is the "bestline region." These subsets can be found efficiently by depth-first search on the powerset of the disks, organized into a suffix tree. Retesting on the crowdsourced test hosts, we found that this algorithm eliminated all of the remaining cases where the predicted region did not cover the true location.

5.2 Effectiveness of Landmarks

To check the observations of Khan et al. [26] and others, that landmarks closer to the target are more useful, we measured the round-trip time between all 250 RIPE Atlas anchors and the target for all of the crowdsourced test hosts. A large majority of all measurements lead to disks that radically overestimate the possible distance between landmark and target. Multilateration produces the same final prediction region even if these overestimates are discarded. We call these measurements *ineffective*. As shown in Figure 11, effective measurements are more likely to come from landmarks close to the target, but among the effective measurements, there is no correlation between distance and the amount by which the measurement reduced the size of the final prediction. This is because a distant landmark may still have only a small overlap with the final prediction region, if it is distant in just the right direction.

5.3 Adaptations for Proxies

When taking measurements through a network proxy, each measured round-trip time is the sum of the RTT from the client to the proxy, and the RTT from the proxy to the landmark. To locate the proxy, we need to measure and subtract the RTT from the client

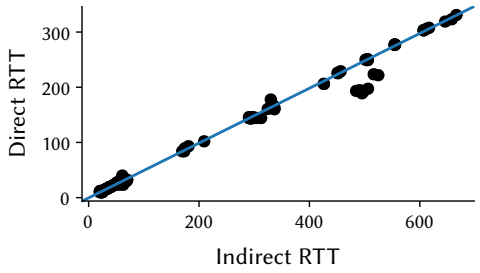

Figure 13: The relationship between direct and indirect round-trip times, η, is almost exactly 1/2.

to the proxy. We cannot measure this directly, because the proxy services usually configure their hosts not to respond to ICMP ping packets, and aggressively rate-limit incoming TCP connections.

Instead, we take inspiration from Castelluccia et al. [5] and have the client ping *itself*, through the VPN, as illustrated in Figure 12. This should take slightly more than twice as long as a direct ping. Figure 13 shows the relationship between direct and indirect pings for all of the proxies in the study that can be pinged both ways. The blue line is a robust linear regression, whose slope η is the inverse of the `RTT_factor` described by Castelluccia et al.. In our case, the slope is 0.49 with $R^2 > 0.99$.

6 LOCATING VPN PROXIES

We used the two-phase, proxy-adapted CBG++ to test the locations of proxies from seven VPN providers. This paper's purpose is not to call out any specific provider for false advertising, so we are not naming the seven providers that we tested; however, figure 14 shows their rankings by number of countries and dependencies claimed, with 150 of their competitors for comparison. Providers A through E are among the 20 that make the broadest claims, while F

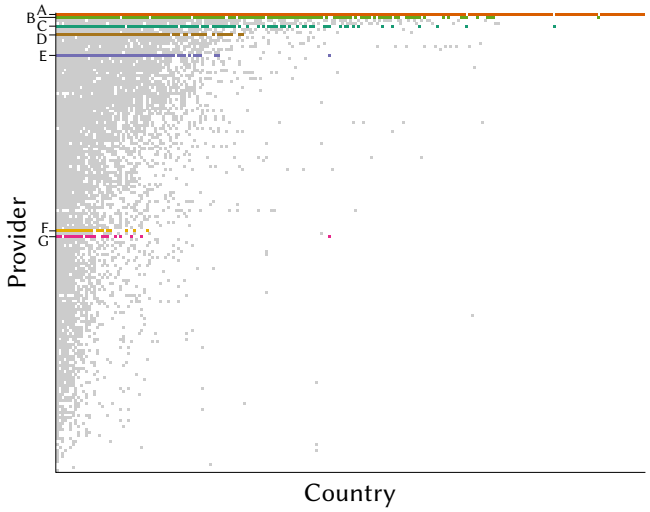

Figure 14: The countries where 157 VPN providers claim to have proxies. Providers included in this study are colored and labeled. Data provided by VPN.com [17].

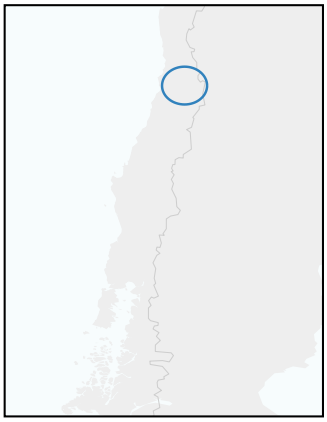

Figure 15: Disambiguation by data center locations: the only data centers in this region are in Chile, not Argentina.

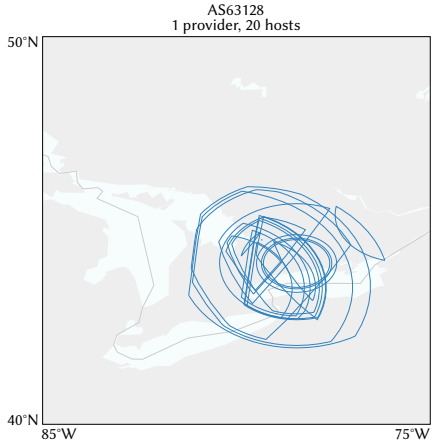

Figure 16: Disambiguation by metadata: all these hosts belong to the same provider, the same AS, and the same /24, so they are likely to be in the same physical location.

and G make more modest and typical claims. Notice that providers who claim only a few locations, tend to claim more or less the same locations; this is what one would expect if it were much easier to lease space in a data center in some countries than others.

All of the VPN providers we tested use round-robin DNS for load balancing; to avoid the possibility of unstable measurements, we looked up all of the server hostnames in advance, from the same host that would run the command-line measurement tool, and tested each IP address separately. We used a single client host for all of the measurements, located in Frankfurt, Germany. Because of this, we cannot say whether the VPN providers might be using DNS geotargeting or anycast routing to direct clients in different parts of the world to different servers. In total, we tested 2269 unique server IP addresses, allegedly distributed over 222 countries and territories.

None of the providers advertise exact locations for their proxies. At best they name a city, but often they only name a country. City claims sometimes contradict themselves; for instance, we observed a config file named "`usa.new-york-city.cfg`" directing the VPN client to contact a server named "`chicago.vpn-provider.example`." Therefore, we only evaluate country-level claims.

CBG++ tells us only that a proxy is within some region. If that region is big enough to cover more than one country, we can't be certain where the server really is. However, we might still be certain that it *isn't* where the proxy provider said it was; for instance, a predicted region that covers Canada and the USA still rules out the entire rest of the world. We say that the provider's claim for a proxy is *false* if the predicted region does not cover any part of the claimed country. We say that it is *credible* if the predicted region is entirely within the claimed country, and we say that it is *uncertain* if the predicted region covers both the claimed country and others. For false and uncertain claims, we also checked whether any of the countries covered by the prediction region were on the same continent as the claimed country.

Some uncertain predictions can be resolved by referring to a list of known locations of data centers, such as the one maintained by the University of Wisconsin [43]. For example, the prediction shown in Figure 15 is uncertain because it covers Argentina as well as Chile. However, the only data centers within the region are in Chile, so we can conclude that this server is in Chile. When data center locations are not enough, cross-checking with network metadata may help. For example, in Figure 16, the largest of the 20 predicted regions cover data centers on both sides of the USA-Canada border, but all of the hosts share a provider, an autonomous system (AS), and a 24-bit network address, which means they are practically certain to be in the *same* data center. Since all of the regions cover part of Canada, but only some of them cross into the USA, we ascribe all of these hosts to Canada. Overall, these techniques allow us to reclassify 353 uncertain predictions as credible or not-credible.

Putting it all together, we find that the claimed location is credible for 989 of the 2269 IP addresses, uncertain for 642, and false for 638. For 401 of the false addresses, the true location is not even on the same continent as the claimed location; however, for 462 of the uncertain addresses, the true location *is* somewhere on the same continent as the claimed location. (See Appendix A for how we defined continental boundaries, and some discussion of which countries and continents are most likely to be confused.)

Figure 17 shows which countries, overall, are more likely to host credibly-advertised proxies, and where the servers for the false claims actually are. The ten countries with the largest number of claimed proxies account for 84% of the credible cases, and only 11% of the false cases. (Uncertain cases are nearly evenly split between the top ten and the remainder.) False claims are spread over the "long tail" of countries, with only a few advertised servers each. Figure 17 also shows the overall effect of using data center and AS information to disambiguate predictions. It is particularly effective when the prediction region crosses continents; 55% of those cases were completely resolved for our purposes. Only 23% of the regions covering multiple countries within the same continent could be disambiguated.

Figure 18 shows another perspective on the same observation, by relating credibility to the country ranking in Figure 14. The credible claims are concentrated in the countries where many other VPN providers also claim to host proxies. This is evidence for our

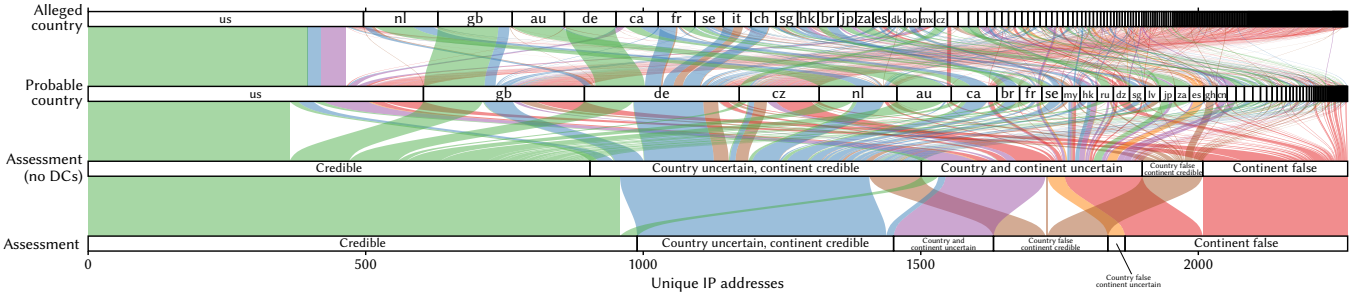

Figure 17: Overall assessment of providers' claims to have proxies in specific countries.

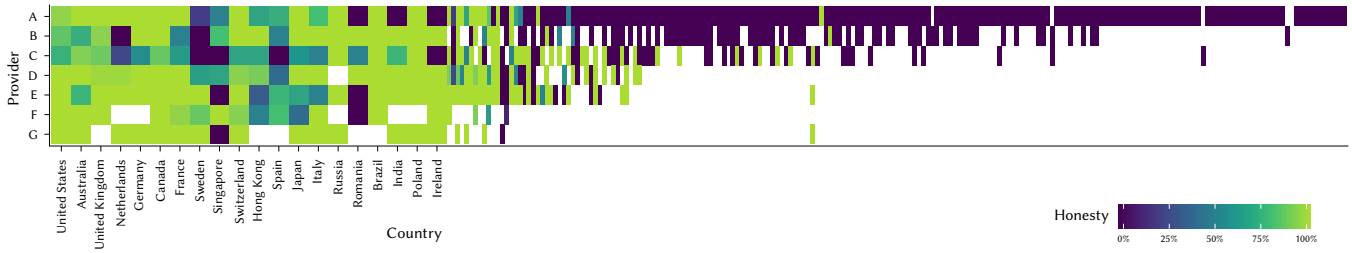

Figure 18: Credible claims are concentrated in the most commonly claimed countries.

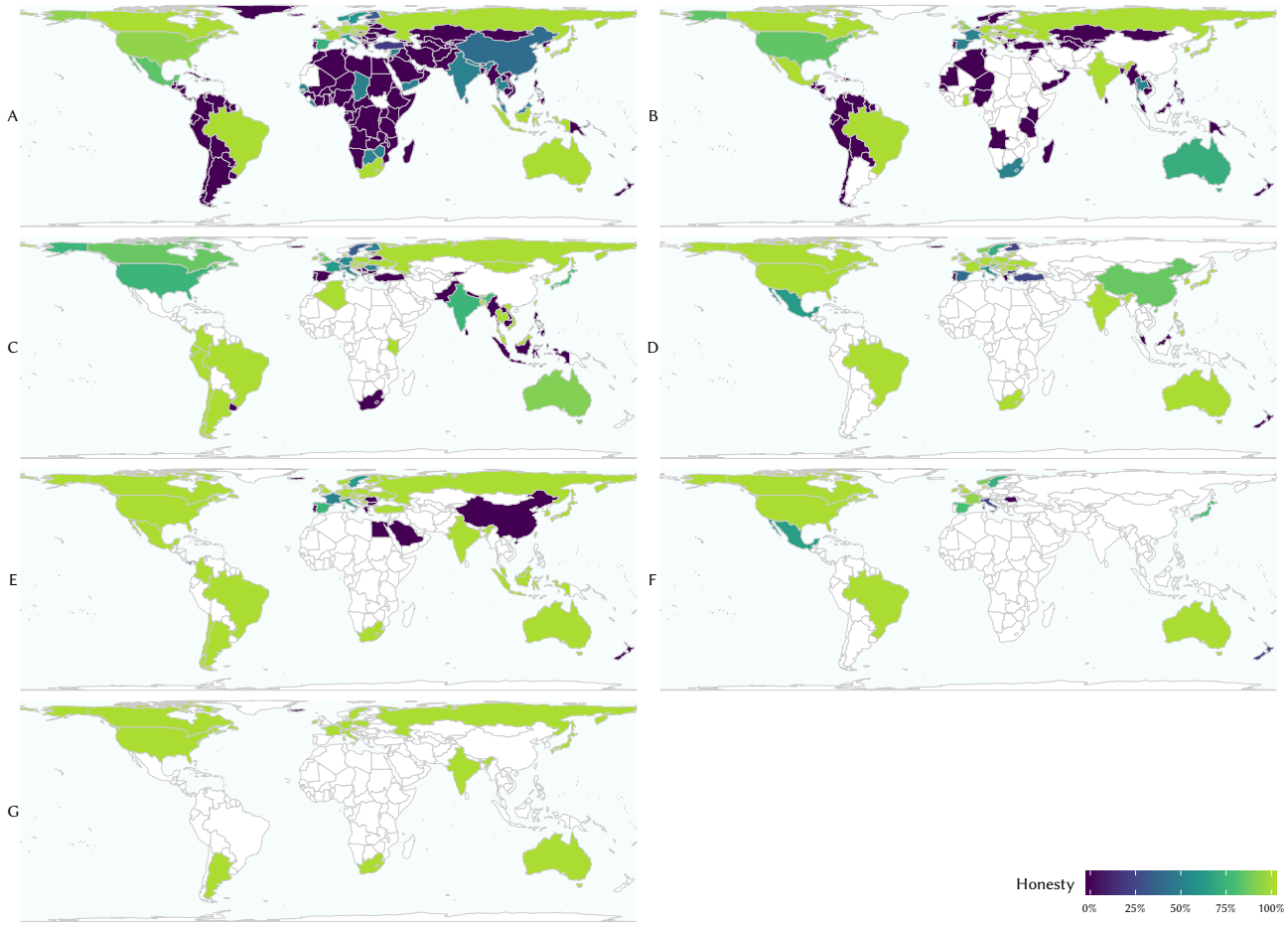

Figure 19: The credibility of each provider's claims for specific countries.

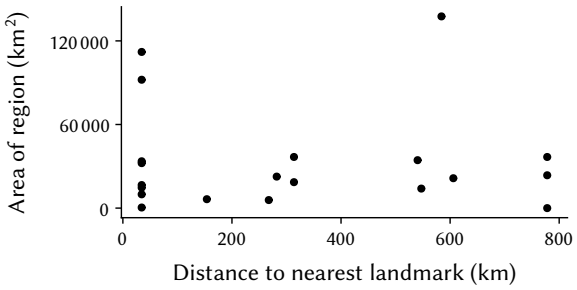

Figure 20: For AS63128, the size of the prediction region is not correlated with the distance to the nearest landmark.

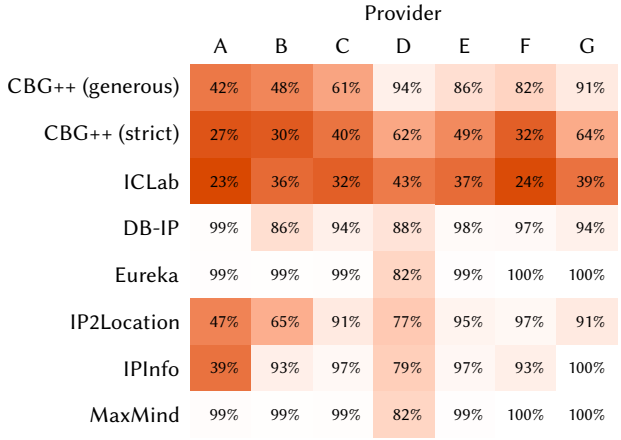

Figure 21: The percentage of each provider's proxies for which our validation (two different ways), ICLab's validation, and five popular geolocation databases agree with the advertised location.

original intuition that proxies are likely to be hosted in countries where server hosting is easy to acquire.

We might also like to know if some providers are more reliable than others. Figure 19 shows, for each provider, a map of the world with each country color-coded according to the overall honesty of the provider's claims for that country. If a country is drawn in white, the provider didn't claim to have any proxies in that country to begin with. Bright green means all of the claimed proxies' CBG++ predictions overlap the country at least somewhat—that is, the "yes" or "uncertain" categories in Figure 17, after taking data center locations into account. Dark purple means none of the predictions overlap the country at all. Colors in between mean CBG++ backs up the claim for some but not all of the proxies claimed to be in that country.

There is some variation among the providers; for instance, C and E are actually hosting servers in more than one country of South America, whereas providers A and B just say they are. However, claimed locations in countries where server hosting is difficult are almost always false. Even in regions like Western Europe, where hosting *is* available in any country one would like, providers seem to prefer to concentrate their hosts in a few locations.

6.1 Data Centers and Prediction Error

Groups of proxies that we believe are all in the same data center can be used for another check on the accuracy of our geolocation methods. We are not yet certain enough of our data center groups to run this analysis on all of the grouped proxies, but we can discuss the results for a clear-cut case like AS63128. If geolocation worked perfectly, all of the regions shown in Figure 16 ought to be the same, but clearly they are not; there isn't even a single sub-region that they all cover. Since our two-phase procedure uses a different randomly-selected group of landmarks for each measurement, variation is to be expected. Figure 20 shows that there is no correlation between the size of each prediction region, and the distance to the nearest landmark for that region from the centroid of all the predictions taken together. This means the variation is not simply due to geographic distance; it may instead be due to some landmarks experiencing more congestion or routing detours than others.

6.2 Comparison with ICLab and IP-to-Location Databases

Active geolocation has been applied to proxy servers once before, as part of a larger project, ICLab, to automate monitoring for Internet censorship across the entire world [39]. We contacted them and they provided us with their data for comparison.

ICLab's geolocation checker only attempts to prove that each proxy is *not* in the claimed country. It assumes that it is impossible for a packet to have traveled faster than a configurable speed limit; their actual tests used 153 km/ms (0.5104 c) for this limit (slightly faster than the "speed of internet" described in Katz-Bassett et al. [25]). Given a country where a host is claimed to be, and a set of round-trip measurements, ICLab's checker calculates the minimum distance between each landmark and the claimed country, then checks how fast a packet would have had to travel to cover that distance in the observed time. The claimed location is only accepted if none of the packets had to travel faster than the limit.

Figure 21 shows the percentage of overall claims by each proxy provider that our algorithm, ICLab's algorithm, and five popular IP-to-location databases agree with. The numbers for CBG++ are calculated two ways: "generous" means we assume that all of the "uncertain" cases are actually credible, and "strict" means we assume they are all false. ICLab's algorithm is even stricter than ours, but most of that is explained by our more subtle handling of uncertain cases. Our "strict" numbers are usually within 10% of ICLab's. Looking more deeply into the disagreements reveals that CBG++ almost always predicts a location close to a national border—just the situation where either algorithm could be tripped up by an underestimate.

All five of the IP-to-location databases are more likely to agree with the providers' claims than either active-geolocation approach is. As discussed earlier, we are inclined to suspect that this is because the proxy providers have influenced the information in these databases. We have no hard evidence backing this suspicion, but we observe that there is no pattern to the countries for which the

IP-to-location databases disagree with provider claims. This is what we would expect to see if the databases were being influenced, but with some lag-time. As the proxy providers add servers, the databases default their locations to a guess based on IP address registry information, which, for commercial data centers, may be reasonably close to the truth. When the database services attempt to make a more precise assessment, this draws on the source that the providers can influence.

7 RELATED WORK

Existing "measurement networks" such as PlanetLab [37], RIPE Atlas [41], or CAIDA Ark [6] have poor coverage outside Europe and North America, and at their best they only offer enough landmarks for city-scale geolocation. Wang et al. [44] propose to increase landmark density to the point where "street-level" geolocation is feasible, by enlisting small businesses' Web servers as additional landmarks, on the assumption that each server is physically located at the street address of the business. They apply heuristics to exclude sites on shared hosting and centralized corporate networks. Chen et al. [8] improves on this by using constrained mean-square error optimization to validate and fine-tune the location of each street-level server.

As we mentioned in Section 2, researchers in this field have devoted considerable effort to modeling the minimum and maximum feasible distance for each round-trip time measurement. Another line of research involves incorporating other sources of information as well as end-to-end round-trip time measurements. The original Octant (not the reimplementation in this paper) assumes that the target's LAN is probably small and any time spent within it is pure overhead, so it uses traceroute information to subtract off elapsed time up to the point where the routes begin to diverge. Komosny et al. [27] elaborated on this by using the Vivaldi [9] "network coordinate system" to model all of the observed distances between intermediate routers, but did not find an improvement over CBG. On the other hand, TBG [25] reports substantial improvements over CBG by using constrained optimization to do the same thing.

Eriksson et al. [15] recommends taking into account prior knowledge (in the Bayesian sense) about where a target host can plausibly be, such as geography ("must be on land"), population density ("more likely to be in a large city"), or known locations of data centers. Alidade [7] builds on this concept, drawing on both active measurements and passive data sources to compile a database that can be queried as easily as a traditional IP-to-location database, but with improved accuracy. OpenIPMap [10] has similar goals and also makes use of crowdsourced location reports.

8 DISCUSSION

We have demonstrated the viability of a simple algorithm for active geolocation, CBG++, at global scale, especially when it is possible to use a crude location estimate to select landmarks within the same continent as the target. We have also confirmed that it is possible to geolocate proxy servers, even when they cannot be directly pinged. Our implementation of the four geolocation algorithms, as well as our measurement code, is publicly available at https://github.com/zackw/active-geolocator.

We have also put to the test the location claims of seven major commercial proxy operators. Our findings are dire: advertised server locations cannot be relied upon, especially when the operators claim to have servers in locations where server hosting is difficult. At most 70% of the servers are where their operators say they are, and that is giving them the full benefit of the doubt; we can only confidently confirm the providers' claims for about 50% of the servers, and all of those are in countries where hosting is easy. Provider A is especially misleading, but all seven of the providers we evaluated had at least a couple of questionable hosts. We shared our results with the providers and asked for an explanation, but all of them declined to respond.

Our results call into question the validity of any network measurement that used VPNs to gain location diversity, especially to diversify beyond Europe and North America. Also, despite a steady stream of reports that IP-to-location databases are unreliable (e.g. [18, 38, 42]) they are still relied upon in numerous contexts; we add our voices to those earlier notes of caution.

As we mentioned in the introduction, many of a VPN provider's customers might be content to appear to be in a specific country. We are not aware of anyone having investigated what VPN customers think they are buying, when they subscribe to a provider that advertises servers in many countries. It would be interesting to find out. Relatedly, while it is well-known that commercial IP-to-location databases contain *errors*, we are not aware of anyone having investigated the possibility of their containing deliberately false information (perhaps because the database compilers themselves were deceived).

One might also wonder whether the VPN operators could actively mislead investigators about the true location of their servers, by interfering with round-trip time measurements. They have no particular reason to do this now, but if active location validation becomes common, they might be motivated to try it. Previous work has found that hostile geolocation targets can indeed foul a position estimate. Gill et al. [19] and others [2, 33] report that *selective* added delay can displace the predicted region, so that its centroid is nowhere near the target's true location; more sophisticated delay-distance models are more susceptible to this, especially if they derive minimum as well as maximum feasible distances from delay measurements. Abdou et al. [3] go further, describing two methods for modifying ICMP echo replies so that some landmarks compute *smaller* round-trip times than they should; with this ability, an adversarial target can shift the predicted region to be anywhere in the world, irrespective of its true location.

Our measurements use TCP handshakes, which include anti-forgery measures, rather than ICMP echo exchanges; also, we can trust both the landmarks and the host running the measurement tool. It is the VPN proxy, in the middle, that is the target of geolocation and not trusted. Unfortunately, being in the middle means it is *easier* for a proxy target to manipulate RTTs both up and down, than it was for an end-host target as considered by Abdou et al. It can selectively delay packets, and it can also selectively forge early SYN-ACKs without needing to guess sequence numbers, since it sees the SYNs. Conceivably, we could prevent this by using landmarks that report their own idea of the time, unforgeably, e.g. authenticated NTP servers [13]—if we could be sure that our measurement client

and all of the landmarks already had synchronized clocks, which is a substantial engineering challenge in itself.

Finally, our Web-based measurement technique could be used to geolocate any visitor to a malicious website without their knowledge or consent. This would be foiled by the use of a proxy, VPN, Tor, or similar, in much the same way that IP-based geolocation is foiled by masking one's IP address with these tools. However, it is still an argument against allowing Web applications to record high-precision information about page load timings, and we plan to discuss this with the major browser vendors.

8.1 Future work

We were only able to include seven VPN providers in this study; there are at least 150 others, some of which make claims nearly as extravagant as provider A. We intend to expand the study to cover as many additional providers as possible, in cooperation with researchers and consumer watchdog organizations looking into other ways commercial VPN providers may fail to live up to their users' expectations. This will also allow us to repeat the measurements over time, and report on whether providers become more or less honest as the wider ecosystem changes.

In order to understand the errors added to our position estimates by the indirect measurement procedure described in Section 5.3, we are planning to set up test-bench VPN servers of our own, in known locations worldwide, and attempt to measure their locations both directly and indirectly.

While our two-phase measurement process is fast and efficient, it also produces noisy groups of measurements like those shown in Figure 16. We think this can be addressed with an iterative refinement process, in which additional probes and anchors are included in the measurement as necessary to reduce the size of the predicted region.

We are experimenting with an additional technique for detecting proxies in the same data center, in which we measure round-trip times to each proxy from each other proxy. Pilot tests indicate that some groups of proxies (including proxies claimed to be in separate countries) show less than 5 ms round-trip times among themselves, which practically guarantees they are on the same local network.

It would be valuable to have a measurement tool that is as user-friendly as the existing Web-based tool, but as accurate as the command-line tool. The Web-based tool could reliably record the time for a single round-trip, and perhaps also avoid some of the Windows-specific overhead and noise, if it could use the W3C Navigation Timing API [1]. This API gives Web applications access to detailed information about the time taken for each stage of an HTTP query-response pair. Unfortunately, it can only be used if each server involved allows it, and currently none of the RIPE Atlas anchors and probes do. We plan to discuss the possibility with the RIPE team. Of course, as we mentioned above, the fact that active geolocation from a Web application is possible at all arguably constitutes a privacy leak in Web browsers, and we also plan to discuss that with the browser vendors.

RIPE Atlas anchors tend to be on subnetworks with more stable, less congested connectivity to the global backbone than is typical for their locale. That could mean each anchor's CBG++ bestline, calibrated from measurements of round-trip times to the other anchors, overestimates the distance packets can typically travel from that anchor. Overestimation leads only to greater uncertainty in predicted locations, whereas underestimation leads to failure (as discussed in Section 5.1). Still, this is a source of error that should be quantified. We are considering adding other measurement constellations, such as the CAIDA Archipelago [6] and PlanetLab [37], to our landmark set. This would allow us to compare the delay-distance relationships observed across constellations to those observed within a single constellation, and thus investigate the degree of overestimation. Additional constellations would also improve our landmark coverage outside Europe and North America. All of the above are also concentrated in the "developed world," but in sparse networks, each new landmark helps a great deal [16].

ACKNOWLEDGMENTS

We must first acknowledge the 40 anonymous volunteers and 150 anonymous Mechanical Turk workers who provided test data. We also thank Michael Gargiulo of VPN.com and Mohammad Taha Khan of the University of Illinois at Chicago for providing data on the commercial VPN market.

We are grateful to our shepherd, Georgios Smaragdakis, and to all of the anonymous reviewers, for their feedback. Pamela Griffith, Sumana Harihareswara, Arian Niaki, Abbas Razaghpanah, Rachee Singh, Mahmood Sharif, Kyle Soska, and Janos Szurdi also provided helpful comments during the writing of this paper, and helped disseminate our call for volunteers.

This research was partially supported by a fellowship from the Open Technology Fund, and partially supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative Program (IITP-2017-R0346-16-1007) supervised by the IITP (Institute for Information & Communications Technology Promotion).

REFERENCES

[1] 2012. Navigation Timing. W3C Recommendation. http://www.w3.org/TR/2012/REC-navigation-timing-20121217/

[2] AbdelRahman M. Abdou, Ashraf Matrawy, and Paul C. Van Oorschot. 2015. CPV: Delay-based Location Verification for the Internet. *Transactions on Dependable and Secure Computing* 14, 2 (2015), 130–144. https://doi.org/10.1109/TDSC.2015.2451614

[3] AbdelRahman M. Abdou, Ashraf Matrawy, and Paul C. Van Oorschot. 2017. Accurate Manipulation of Delay-based Internet Geolocation. In *Asia Conference on Computer and Communications Security*. ACM, New York, NY, USA, 887–898. https://doi.org/10.1145/3052973.3052993

[4] Mohammed Jubaer Arif, Shanika Karunasekera, and Santosh Kulkarni. 2010. GeoWeight: Internet Host Geolocation Based on a Probability Model for Latency Measurements. In *Australasian Computer Science Conference*, Vol. 102. ACS, Sydney, 89–98. http://crpit.com/confpapers/crpitv102arif.pdf

[5] Claude Castelluccia, Mohamed Ali Kaafar, Pere Manils, and Daniele Perito. 2009. Geolocalization of Proxied Services and its Application to Fast-Flux Hidden Servers. In *Internet Measurement Conference*. ACM, New York.

[6] Center for Applied Internet Data Analysis. 2006. Archipelago (Ark) Measurement Infrastructure. http://www.caida.org/projects/ark/

[7] Balakrishnan Chandrasekaran, Mingru Bai, Michael Schoenfield, Arthur Berger, Nicole Caruso, George Economou, Stephen Gilliss, Bruce Maggs, Kyle Moses, David Duff, Keung-Chi Ng, Emin Gün Sirer, Richard Weber, and Bernard Wong. 2015. *Alidade: IP Geolocation without Active Probing*. Technical Report CS-TR-2015.001. Department of Computer Science, Duke University.

[8] Jingning Chen, Fenlin Liu, Xiangyang Luo, Fan Zhao, and Guang Zhu. 2016. A landmark calibration-based IP geolocation approach. *EURASIP Journal on Information Security* 2016, Article 4 (2016), 11 pages. https://doi.org/10.1186/s13635-015-0029-5

[9] Frank Dabek, Russ Cox, Frans Kaashoek, and Robert Morris. 2004. Vivaldi: A Decentralized Network Coordinate System. In *SIGCOMM*. ACM, New York, 15–26. https://doi.org/10.1145/1015467.1015471

[10] Jasper den Hertog and Massimo Candela. 2018. OpenIPMap: A Collaborative Approach to Mapping Internet Infrastructure. https://labs.ripe.net/Members/jasper_den_hertog/openipmap-a-collaborative-approach-to-mapping-internet-infrastructure
[11] Shichang Ding, Xiangyang Luo, Meijuan Yin, Yan Liu, and Fenlin Liu. 2015. An IP Geolocation Method Based on Rich-Connected Sub-networks. In *International Conference on Advanced Communication Technology*. IEEE, Piscataway, NJ, 176–181. https://doi.org/10.1109/ICACT.2015.7224779
[12] Ziqian Dong, Rohan D. W. Perera, Rajarathnam Chandramouli, and K. P. Subbalakshmi. 2012. Network measurement based modeling and optimization for IP geolocation. *Computer Networks* 56, 1 (2012), 85–98. https://doi.org/10.1016/j.comnet.2011.08.011
[13] Benjamin Dowling, Douglas Stebila, and Greg Zaverucha. 2016. Authenticated Network Time Synchronization. In *USENIX Security Symposium*. USENIX Association, 823–840. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/dowling
[14] Brian Eriksson, Paul Barford, Bruce Maggs, and Robert Nowak. 2012. Posit: A Lightweight Approach for IP Geolocation. *SIGMETRICS Performance Evaluation Review* 40, 2 (2012), 2–11. https://doi.org/10.1145/2381056.2381058
[15] Brian Eriksson, Paul Barford, Joel Sommers, and Robert Nowak. 2010. A Learning-Based Approach for IP Geolocation. In *Passive and Active Measurement*, Arvind Krishnamurthy and Bernhard Plattner (Eds.). Springer, Berlin, Heidelberg, 171–180. https://doi.org/10.1007/978-3-642-12334-4_18
[16] Brian Eriksson and Mark Crovella. 2013. Understanding Geolocation Accuracy using Network Geometry. In *INFOCOM*. IEEE, Piscataway, NJ, 75–79. https://doi.org/10.1109/INFCOM.2013.6566738
[17] Michael Gargiulo. 2018. List of VPN Locations by Provider. In *VPN Reviews & Free Comparison Charts*. https://www.vpn.com/
[18] Manaf Gharaibeh, Anant Shah, Bradley Huffaker, Han Zhang, Roya Ensafi, and Christos Papadopoulos. 2017. A Look at Router Geolocation in Public and Commercial Databases. In *Internet Measurement Conference*. ACM, New York. https://doi.org/10.1145/3131365.3131380
[19] Phillipa Gill, Yashar Ganjali, Bernard Wong, and David Lie. 2010. Dude, where's that IP?: Circumventing measurement-based IP geolocation. In *USENIX Security*. USENIX, Berkeley, CA, Article 16, 16 pages. http://static.usenix.org/legacy/events/sec10/tech/full_papers/Gill.pdf
[20] Bamba Gueye, Artur Ziviani, Mark Crovella, and Serge Fdida. 2004. Constraint-based Geolocation of Internet Hosts. In *Internet Measurement Conference*. ACM, New York, 288–293. https://doi.org/10.1145/1028788.1028828
[21] Kashmir Hill. 2016. How an internet mapping glitch turned a random Kansas farm into a digital hell. *FUSION* (2016). http://fusion.net/story/287592/internet-mapping-glitch-kansas-farm/
[22] Thomas Holterbach, Cristel Pelsser, Randy Bush, and Laurent Vanbever. 2015. Quantifying Interference Between Measurements on the RIPE Atlas Platform. In *Internet Measurement Conference*. ACM, New York, 437–443. https://doi.org/10.1145/2815675.2815710
[23] Zi Hu, John Heidemann, and Yuri Pradkin. 2012. Towards Geolocation of Millions of IP Addresses. In *Internet Measurement Conference*. ACM, New York, 123–130. https://doi.org/10.1145/2398776.2398790
[24] Collin Jackson, Andrew Bortz, Dan Boneh, and John C Mitchell. 2006. Protecting Browser State from Web Privacy Attacks. In *World Wide Web*. ACM, New York, 737–744. http://www.stanford.edu/people/jcm/papers/sameorigin.pdf
[25] Ethan Katz-Bassett, John P. John, Arvind Krishnamurthy, David Wetherall, Thomas Anderson, and Yatin Chawathe. 2006. Towards IP Geolocation Using Delay and Topology Measurements. In *Internet Measurement Conference*. ACM, New York, 71–84. https://doi.org/10.1145/1177080.1177090
[26] Raja A. A. Khan, Anjum Naveed, and R. Les Cottrell. 2016. *Adaptive Geolocation of Internet Hosts*. Technical Report SLAC-PUB-16463. SLAC National Accelerator Laboratory.
[27] Dan Komosny, Milan Simek, and Ganeshan Kathiravelu. 2013. Can Vivaldi Help in IP Geolocation? *Przegląd Elektrotechniczny* 2013, 5 (2013), 100–106. http://www.pe.org.pl/articles/2013/5/20.pdf
[28] Dan Komosny, Miroslav Voznak, Ganeshan Kathiravelu, and Hira Sathu. 2015. Estimation of Internet Node Location by Latency Measurements—The Underestimation Problem. *Information Technology and Control* 44, 3 (2015), 279–286. http://hdl.handle.net/10084/110524
[29] Rupa Krishnan, Harsha V. Madhyastha, Sridhar Srinivasan, Sushant Jain, Arvind Krishnamurthy, Thomas Anderson, and Jie Gao. 2009. Moving Beyond End-to-End Path Information to Optimize CDN Performance. In *Internet Measurement Conference*. ACM, New York, 190–201. https://doi.org/10.1145/1644893.1644917
[30] Sándor Laki, Péter Mátray, Péter Hága, Tamás Sebők, István Csabai, and Gábor Vattay. 2011. Spotter: A Model Based Active Geolocation Service. In *INFOCOM*. IEEE, Piscataway, NJ, 3173–3181. https://doi.org/10.1109/INFCOM.2011.5935165
[31] Raul Landa, Richard G. Clegg, João Taviera Araújo, Eleni Mykoniati, David Griffin, and Miguel Rio. 2013. Measuring the Relationships between Internet Geography and RTT. In *International Conference on Computer Communications and Networks*. IEEE, Piscataway, NJ, 1–7. https://doi.org/10.1109/ICCCN.2013.6614151
[32] Dan Li, Jiong Chen, Chuanxiong Guo, Yunxin Liu, Jinyu Zhang, Zhili Zhang, and Yongguang Zhang. 2013. IP-Geolocation Mapping for Moderately Connected Internet Regions. *Transactions on Parallel and Distributed Systems* 24, 2 (2013), 381–391. https://doi.org/10.1109/TPDS.2012.136
[33] James A. Muir and Paul C. Van Oorschot. 2009. Internet Geolocation: Evasion and Counterevasion. *Comput. Surveys* 42, 1 (2009), 4:1–4:23. https://doi.org/10.1145/1592451.1592455
[34] Péter Mátray, Péter Hága, Sándor Laki, Gábor Vattay, and István Csabai. 2012. On the spatial properties of internet routes. *Computer Networks* 56, 9 (2012), 2237–2248. https://doi.org/10.1016/j.comnet.2012.03.005
[35] Venkata N. Padmanabhan and Lakshminarayanan Subramanian. 2001. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *SIGCOMM*. ACM, New York, 173–185. https://doi.org/10.1145/964723.383073
[36] Tom Patterson, Nathaniel Vaughn Kelso, et al. 2012. Natural Earth. Free vector and raster map data. http://www.naturalearthdata.com/
[37] PlanetLab 2007. PlanetLab: an open platform for developing, deploying, and accessing planetary-scale services. http://www.planet-lab.org/
[38] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. 2011. IP Geolocation Databases: Unreliable? *SIGCOMM Computer Communications Review* 41, 2 (2011), 53–56. https://doi.org/10.1145/1971162.1971171
[39] Abbas Razaghpanah, Anke Li, Arturo Filastò, Rishab Nithyanand, Vasilis Ververis, Will Scott, and Phillipa0 Gill. 2016. Exploring the Design Space of Longitudinal Censorship Measurement Platforms. (2016). arXiv:cs.NI/1606.01979 arXiv preprint.
[40] RIPE NCC Staff. 2015. RIPE Atlas: A Global Internet Measurement Network. *The Internet Protocol Journal* 18, 3 (2015), 2–26. http://ipj.dreamhosters.com/wp-content/uploads/2015/10/ipj18.3.pdf
[41] RIPE Network Coordination Centre. 2014. RIPE Atlas: a global network of Internet probes. https://atlas.ripe.net
[42] Yuval Shavitt and Noa Zilberman. 2011. A Geolocation Databases Study. *Selected Areas in Communications* 29, 10 (2011), 2044–2056. https://doi.org/10.1109/JSAC.2011.111214
[43] University of Wisconsin. 2011. Internet Atlas (DS-468). Continuously updated data set. https://doi.org/10.23721/110/1353976
[44] Yong Wang, Daniel Burgener, Marcel Flires, Aleksandar Kuzmanovic, and Cheng Huang. 2011. Towards Street-Level Client-Independent IP Geolocation. In *Networked Systems Design and Implementation*. USENIX, Berkeley, CA.
[45] Bernard Wong, Ivan Stoyanov, and Emin Gün Sirer. 2007. Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts. In *Networked Systems Design and Implementation*. USENIX, Berkeley, CA, Article 23, 14 pages. http://static.usenix.org/legacy/events/nsdi07/tech/full_papers/wong/wong.pdf
[46] Artur Ziviani, Serge Fdida, José F. de Rezende, and Otto Carlos M. B. Duarte. 2005. Improving the accuracy of measurement-based geographic location of Internet hosts. *Computer Networks* 47, 4 (2005), 503–523. https://doi.org/10.1016/j.comnet.2004.08.013

A UNCERTAINTY AND CONTINENTS

Uncertain prediction regions include more than one country, or even more than one continent. Since a prediction region is always contiguous, we expect uncertainty among groups of neighboring countries, but which groups? We briefly examine this question with a pair of confusion matrices, one for continents and the other for of countries. All data is for the proxies, not the crowdsourced test hosts.

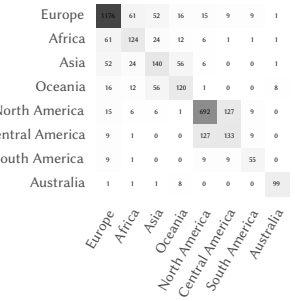

Figure 22: Confusion matrix among continents

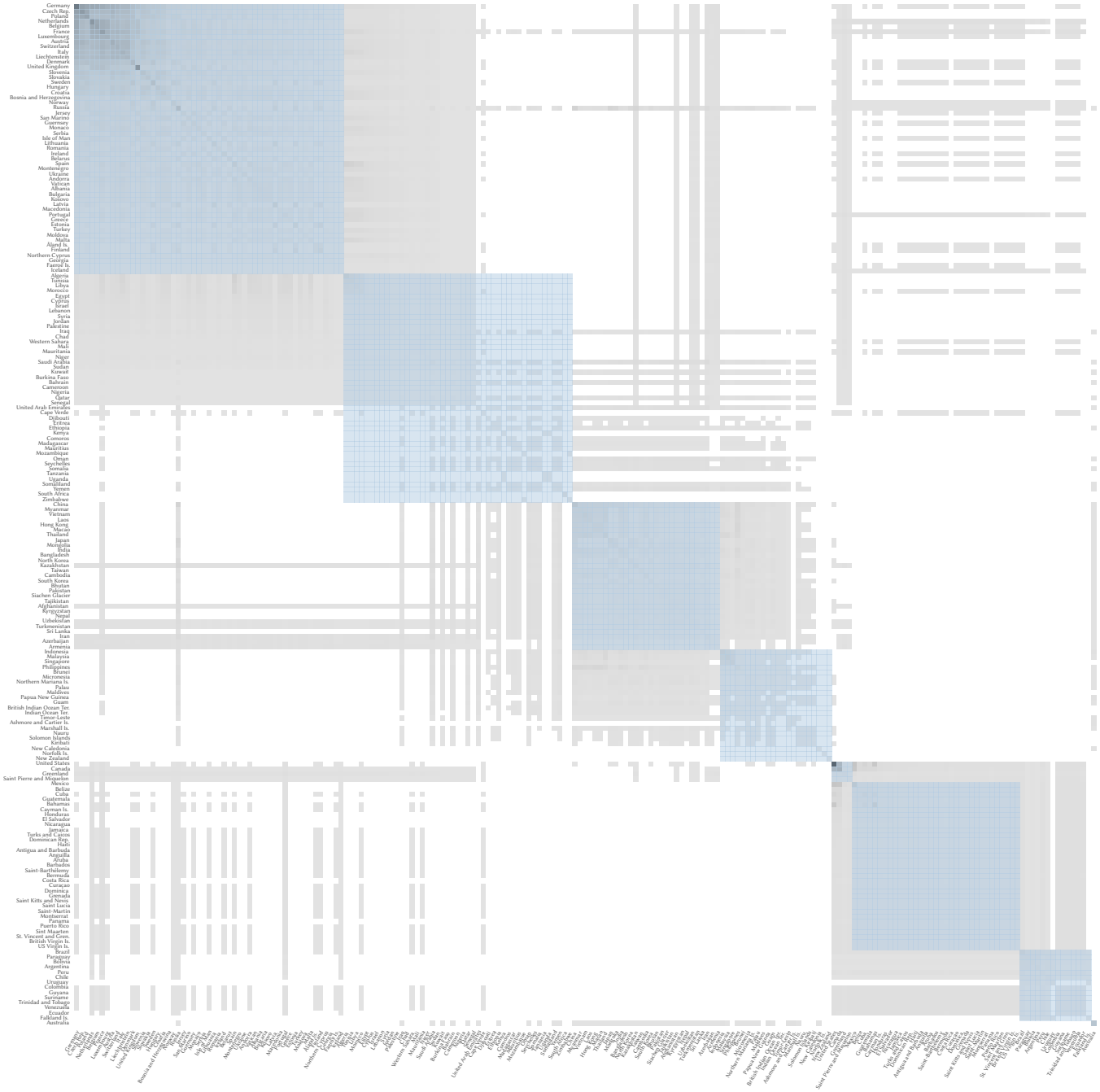

Figure 23: Confusion matrix among countries. Blue tinting marks groups of countries on the same continent.

The lines separating continents are somewhat arbitrary. For this analysis, we chose to include Mexico with Central America, Turkey and Russia with Europe, all of the Middle East with Africa, and all of Malaysia and New Zealand with Oceania.

Intercontinental uncertainty is as one would expect: Europe/Africa/Asia, Asia/Oceania/Australia, North/Central and to a lesser extent South America. The country matrix, however, reveals that just about every country within a continent *can* share a prediction region; the exceptions are more interesting. Many southern African countries seem more likely to be confused with Asia than their neighbors, and not just the Indian subcontinent, but all the way to Japan. Similar effects appear for Oceania. This may reflect neighboring countries or islands in these areas not being connected directly, only through a more developed hub.