TinyG: Accurate IP Geolocation Using a Tiny Number of Probers

Nan Jiang*‡, Jessie Hui Wang*†‡, Jilong Wang*†‡, and Peiran Wang*‡
*Institute for Network Sciences and Cyberspace, Tsinghua University, China
†Zhongguancun Laboratory, Beijing, China
‡Beijing National Research Center for Information Science and Technology

***Abstract*—IP geolocation is essential for various applications. However, the reliability of IP geolocation databases has been proven to be inadequate. In recent years, the growing number of public probers has offered the potential for more accurate geolocation results through active measurement. The conventional practice is to probe the target IP address using all available probers and feed the measurement results to the active geolocation method. However, this practice is cost-inefficient and may trigger the anti-flood mechanism. Moreover, public probers typically impose user-level limits on the frequency and quantity of measurements. Therefore, it is important to reduce the average number of probers (ANP) selected for successfully probing each target. Researchers have discovered that geolocation accuracy primarily depends on the minimum delay between probers and the target. Inspired by that, we propose TinyG, a prober selection algorithm designed to reduce the ANP needed to find probers within a sufficiently small delay from the target. TinyG divides the probing process into multiple rounds and leverages previous measurement results to guide the selection of probers for subsequent rounds. Experimental results show that when the associated minimum delay is within 2 ms, various active geolocation methods can provide credible geolocation results. TinyG outperforms other algorithms in reducing the ANP needed to obtain credible results. Compared to using more than 1,300 probers, TinyG can achieve an ANP of 6.7 with only a 6% coverage loss of credible results.**

***Index Terms*—IP Geolocation, Public Probers, Delay Measurement**

I. Introduction

IP geolocation aims to map the target IP address to the physical location. It has numerous applications across various industries. One prominent use case is in advertising [1], where it enables the delivery of targeted ads based on users' locations, thereby enhancing the effectiveness of marketing campaigns. Additionally, IP geolocation facilitates content localization and regional restrictions, ensuring compliance with local laws and regulations [2, 3]. It also plays a vital role in network security fields such as tracking the source of attacks and detecting BGP threats [4]. Moreover, Internet service providers (ISPs) benefit from IP geolocation for network optimization and management [5–7]. Credible IP geolocation results are essential for these applications. Currently, there are many free or commercial IP geolocation databases, such as MaxMind [8], which are often used for geolocating IP addresses. However, they have been found to have significant discrepancies between their stated accuracy and actual accuracy, as revealed by numerous studies [1, 9–11]. These studies highlight the pressing need to improve the accuracy of IP geolocation results.

Besides using databases, researchers have proposed various active IP geolocation methods, such as CBG [12]. These methods use all available probers to measure the round-trip delay between the probers and the target. Probers are servers with known locations that can send packets to the target IP address. Since the delay is related to the geographical distance, active geolocation methods use different ways to estimate the target's location based on the delays between the target and probers. Researchers [13–15] have demonstrated that geolocation accuracy of these methods is primarily determined by the minimum delay between probers and the target. Accurate CBG requires at least 1 prober to be within a sufficiently small delay from the target, necessitating the prober to be geographically close to the target. Trammell *et al.* [13] described this situation as "lucky". Ben *et al.* [14] have shown that a delay less than 2 ms between the target and a certain prober can provide approximately 95% IP geolocation accuracy at city-level. To achieve a high success rate of finding probers within 2 ms, probers distributed around the world are needed. In recent years, the growing number of public probers has increased this success rate. Most of them are hosted by volunteers around the world, such as Looking Glass (LG) [16] and RIPE Atlas [17]. Currently, there are more than 1,000 LG probers [18] and more than 20,000 RIPE Atlas probers available. Considering the high cost of deploying and maintaining a large number of probers, using public probers is the only viable option for most individuals and organizations to obtain accurate IP geolocation results.

While using all available probers to probe the target can provide the best accuracy, it does not scale in practice [14, 15] as it often triggers anti-flood and DDoS alerts [19] of the firewalls, resulting in response loss. Moreover, public probers typically impose user-level limits on usage to prevent excessive resource consumption [16]. For instance, Hurricane Electric [20] sets a 1-minute forced interval between measurements for each user. To conduct a measurement using a prober of the RIPE Atlas, the user consumes a certain number of credits, which limits the total number of measurements the user can conduct. Given these limits, simply using all available public probers to probe the target is impractical. Thus, it is necessary to reduce the average number of probers (ANP) selected for successfully probing each target, preferably from thousands to a tiny number.

Inspired by the observation that the accuracy is primarily determined by the minimum delay between probers and the target [13, 14], we propose a multi-round prober selection algorithm called TinyG. The objective of TinyG is to reduce the ANP needed for finding a prober within a sufficiently small delay from the target. To achieve this, TinyG divides the process of probing into multiple rounds, utilizing the measurement results from previous rounds to guide prober selection in subsequent rounds. At the beginning of each round, 1 prober is strategically selected to probe the target to measure the delay between them. Subsequently, TinyG calculates the probability distribution of the target's location and uses information entropy to quantify the uncertainty associated with the target's location. TinyG strategically selects a prober for the next round. The loop continues until the measured delay is within 2 ms or the number of rounds exceeds a preset threshold. Experimental results validate that credible geolocation results can be obtained using various active geolocation methods when the minimum delay is within 2 ms and TinyG surpasses other prober selection algorithms in reducing the ANP needed for obtaining credible results.

The rest of this paper is structured as follows: Section II provides the necessary background and discusses related work. In Section III, we present the design of TinyG. The experimental results are presented in Section IV, and the limitations of TinyG are discussed in Section V. Finally, Section VI concludes the paper.

II. Background and Related Work

A. IP Geolocation Methods

In recent decades, numerous IP geolocation methods have been proposed, which can be divided into two categories: information extraction-based methods and active measurement-based methods.

Information extraction-based methods derive location information from specific sources, such as WHOIS and the Domain Name System (DNS), for the purpose of geolocating IP addresses. WHOIS contains IP registration data, including city and country details. An early example is the NetGeo database developed by Moore et al. [21], which utilized WHOIS location information. However, NetGeo's accuracy is poor due to delayed updates in WHOIS and the fact that IP addresses may not be used in their registered locations. For the convenience of operations, the network operators of ISPs often assign DNS hostnames (PTR records) to router interface IP addresses, embedding geographical hints within them to indicate physical locations. For example, the hostname `ae-5.r22.`**`miamfl`**`02.`**`us`**`.bb.gin.ntt.net` indicates that the corresponding IP address is located in Miami, FL, US. Due to the complexity and diversity of embedding rules of different ISPs, researchers have proposed various methods to extract location information from hostnames. Spring et al. [22] manually constructed rules for different ISPs and released the undns tool, suffering from limited coverage and significant labor costs. Other methods, such as DRoP [23] and HLOC [24], automatically extract location information by matching against large geographic information dictionaries like GeoNames [25]. However, they encounter challenges when ISPs deviate from common dictionaries and create their own geographical hints. To address this, Luckie et al. [26] developed Hoiho, which employs automatic rule learning. Their comparison results demonstrated that Hoiho achieves the highest coverage. Besides WHOIS and DNS, location information displayed on web pages can be leveraged. Guo et al. [27] extracted location-related strings from web pages and correlated them with the IP addresses of the hosting servers. Li et al. [28] used the IP addresses and the coordinates of public live webcams as landmarks. Information extraction-based methods do not require conducting active measurements using probers. However, they heavily rely on specific data sources and are unable to geolocate arbitrary IP addresses.

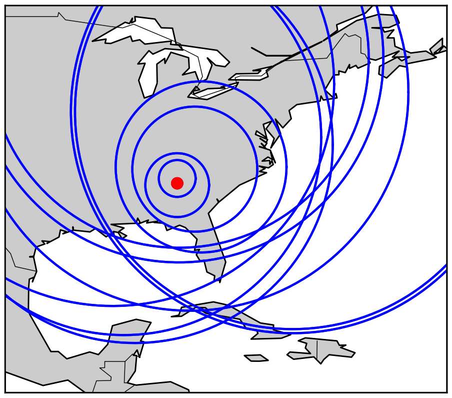
Fig. 1: Exclusion circles correspond to the delays between probers and the target (red dot), with the upper bound distance as the radius. Typically, the larger exclusion circles completely cover the smaller ones.

Active measurement-based methods primarily rely on network delay. The simplest approach, known as Shortest Ping (SP) [29], assigns the location of the prober with the smallest delay directly to the target. The more common approach is to construct exclusion circles using the upper bound of distances corresponding to delays. An exclusion circle is drawn for each prober, with the prober as the center and the upper bound distance corresponding to the delay as the radius. Researchers have proposed various methods to establish upper bounds due to the complex relationship between delay and distance. The most relaxed and safest upper bound is based on the maximum propagation speed of packets (V_{packet}). Previous studies [30, 31] have shown that V_{packet} is approximately $2/3 \cdot V_{light} = 200km/s$, where V_{light} represents the speed of light in vacuum. For simplicity, we refer to it as the *maxspeed-constraint* in this paper. To establish stricter upper bounds, Gueye *et al.* [12] proposed CBG, which uses a linear function to fit the upper bound based on previously collected delay-distance data and geolocates the target to the centroid of the intersection of these circles. In 2017, Trammell *et al.* [13] demonstrated the significant impact of the minimum delay observed during measurements on accuracy. Fig. 1 illustrates an example where exclusion circles of small delays are typically covered by those of larger delays, and thus the intersections of the larger circles have no contribution to the result, indicating that the accuracy is mainly determined by the minimum delay. Laki *et al.* [32] developed Spotter, a probability-based geolocation method that models the probability density function of distance at a given delay using a Gaussian distribution. Spotter calculates the probability distribution of the target's location based on delay measurements and assigns the target to the region with the highest probability. Additionally, some researchers have explored combining delay and topology to geolocate IP addresses appearing in routing paths. Katz *et al.* [29] performed traceroute from probers to a large number of targets and used global optimization to estimate the locations of IP addresses appearing in the traceroute results. Tian *et al.* [33] geolocated China's Internet routers' IP addresses based on the hierarchy of topology. Dan *et al.* [34] geolocated routers' IP addresses by grouping together two IP addresses with a small link delay.

B. Prober Selection Algorithms

Previous work [12, 29, 32, 35] typically used as many probers as possible to conduct measurements for each target, which is impractical when dealing with a large number of available probers. To address this challenge, Hu et al. [15] proposed a prefix-based prober selection algorithm to reduce the ANP. [15] leverages the observation that IP addresses within the same /24 prefix are typically located in close proximity. It selects some IP addresses from each /24 prefix as the representatives and probes them using **all** available probers. For the remaining IP addresses, only the 10 probers closest to the representatives of the corresponding /24 prefix are selected for probing. Although [15] reduces the ANP by selecting a few probers that are likely to be closest to the target, it has the following limitations:

- It fails to select probers that are close to the target when the IP addresses within a /24 prefix are not geographically clustered. This issue arises when two routers from different cities, countries, or continents are interconnected, with the interface IP addresses on the link belonging to the same /24 prefix while located in very different areas [6].
- It cannot effectively reduce the ANP when the given target IP address list is **sparse**, meaning that the number of targets within the same /24 prefix is small while the total number of targets is large. This limitation stems from the fact that it initially uses all probers to probe the representatives of each prefix.

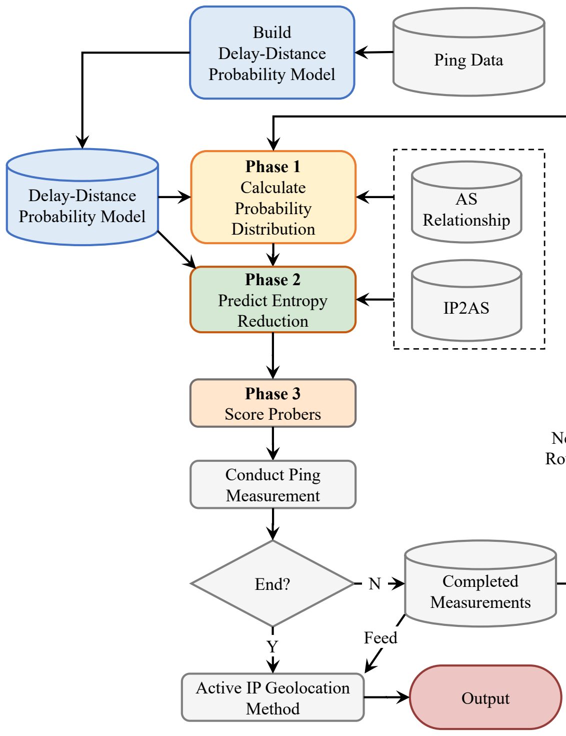

Fig. 2: TinyG selects 1 prober to probe the target in each round based on the results of previous rounds. Finally, the measurement results of all rounds are fed to the active geolocation method.

Unlike [15], TinyG selects probers for each target separately. reducing the ANP from thousands to a tiny number, regardless of the distribution of the target IP address list and device types.

III. Design of TinyG

A. Overview

Different from previous work, TinyG splits the probing process for a target into multiple rounds. In each round, TinyG strategically selects 1 prober to measure the delay between the prober and the target. The delay threshold 2 ms and the max number of rounds $\mathcal{R}$ control the termination. If the measured delay is within 2 ms or the number of rounds exceeds $\mathcal{R}$, TinyG stops probing and then feeds the measurement results of all rounds to an active measurement-based IP geolocation method like SP, CBG, or Spotter.

At the end of each round, TinyG estimates the target's location by calculating its probability distribution and quantifies the uncertainty through information entropy. To illustrate the relationship between probability density distribution and entropy, we sequentially use probers located in Frankfurt, Berlin, Paris, and London to probe an IP address located in Amsterdam, and the results are shown in Fig. 3. In round 1 and round 2, the large high-probability region makes estimating the target's location challenging. As more measurements are conducted, the entropy decreases and the high-probability region **contracts**, enabling easier finding of the prober within

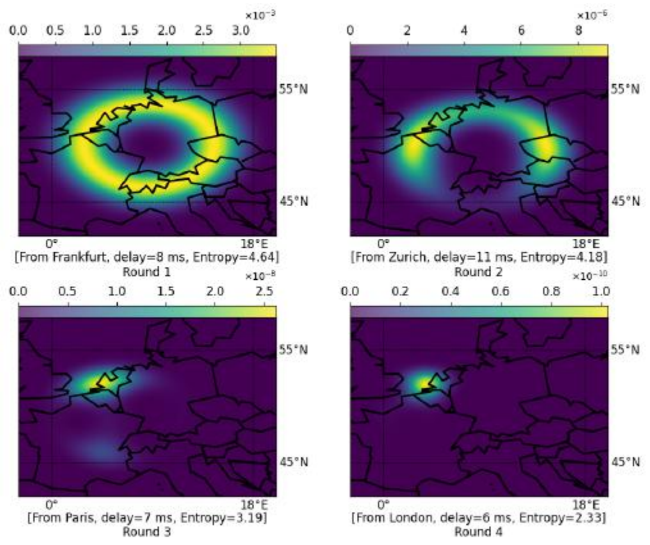
[From Frankfurt, delay=8 ms, Entropy=4.64]
Round 1
[From Zurich, delay=11 ms, Entropy=4.18]
Round 2
[From Paris, delay=7 ms, Entropy=3.19]
Round 3
[From London, delay=6 ms, Entropy=2.33]
Round 4
55°N
45°N
0°
18°E

Fig. 3: The probability density and the corresponding entropy based on completed measurement results in different rounds.

2 ms from the target. TinyG adjusts prober selection based on the level of uncertainty and the number of round. Instead of switching strategies according to a fixed threshold of entropy, TinyG implements a dynamic scoring mechanism. Higher scores are assigned to probers that contribute to greater entropy reduction on average when entropy is high or the number of round is small, while probers located in the high-probability region receive higher scores when entropy is low or the number of round is large.

To facilitate the calculation of the probability distribution and information entropy, we build a delay-distance probability model in advance (Section III-B). To select the prober for the next round, TinyG goes through 3 phases, as depicted in Fig. 2. Phase 1 (Section III-C) involves the caculation of the target's probability distribution and information entropy based on the measurement results. In Phase 2 (Section III-D), TinyG predicts the expected reduction in entropy resulting from different probers in the next round. Finally, in Phase 3 (Section III-E), TinyG uses a dynamic scoring mechanism to assign scores to the probers and selects the prober with the highest score for the next round.

B. Build Delay-Distance Probability Model

To calculate the probability distribution, we require a probability density function that describes the likelihood of the target being at a given distance from the prober for a given delay. Additionally, in order to predict entropy reduction for each prober in the next round, we need to estimate the delay at a given distance from the target. Therefore, we build a delay-distance model offline using previously collected delay-distance data. The model consists of two parts: one part outputs the probability density of the target at a given distance from the prober for a given delay, while the other part estimates the delay at a given distance from the target. For the first part, we refer to the calculation process in Spotter [32]. Additionally, we observe that the relation between delay and distance is influenced by the Autonomous System (AS) relations between probers and targets. We categorize the relations between ASes into 3 types: *Same*, *Neighbor*, and *Stranger*. Then, we incorporate the AS relation into the model, which was not considered in Spotter.

Probability Density Function: Suppose we conduct a delay measurement from a prober to the target, resulting in a measured delay d. We denote the distance between the target and the prober as a random variable s. The probability density for s based on delay d and the AS relation a between the prober and the target can be expressed as $f_{a,d}(s)$, indicating the likelihood of the target being at that distance from the prober. To build $f_{a,d}(s)$, we fit the previously collected delay-distance data using a Gaussian distribution. We classify the data according to the AS relation and fit the probability density functions separately.

Delay Estimation: In phase 2, we use the mathematical expectation of delay to predict the entropy reduction of using different probers in the next round. For a given distance s and AS relation a, we denote the mathematical expectation of delay as $S2T(a,s)$. To calculate it, we analyze previously collected data and calculate the average delay for distances within the range of $[80\% \cdot s, 120\% \cdot s]$, as it is difficult to find exact matches for distance s.

C. Phase 1: Calculating Probability Distribution

TinyG utilizes probers world wide, denoted by $\mathbb{V}_i$ (i=1,2,...). In each round, Tiny selects 1 prober to probe the target. We use a 3-tuple ($\mathbb{Y}_t$, $\mathcal{A}_t$, $\mathcal{T}_t$) to denote the measurement result of round t (t=1,2,...,$\mathcal{R}$), wherein $\mathbb{Y}_t$ is the prober selected by TinyG for probing; $\mathcal{A}_t$ is the AS relation between $\mathbb{Y}_t$ and the target; and $\mathcal{T}_t$ is the delay between $\mathbb{Y}_t$ and the target. At the end of each round, TinyG updates the probability distribution of the target being in different cities. The cities where the target could be located are denoted by $\mathcal{C}_k$ (k=1,2,...). Here, we only consider the cities where there are probers and dynamically remove infeasible cities according to *maxspeed-constraint*. We denote the measurement results from round 1 to round t as $\mathcal{M}_t$, and the conditional probability that the target falls into $\mathcal{C}_k$ at the end of round t can be calculated by integrating the probability density:

$$\begin{aligned} P(\mathcal{C}_k|\mathcal{M}_t) &= P(\mathcal{C}_k|\mathcal{M}_{t-1}) \cdot P(\mathcal{C}_k|(\mathbb{Y}_t,\mathcal{A}_t,\mathcal{T}_t)) \\ &= P(\mathcal{C}_k|\mathcal{M}_{t-1}) \int\limits_{\mathcal{C}_k} f_{\mathcal{A}_t,\mathcal{T}_t} S(\mathbb{Y}_t,\sigma) \cdot d\sigma \end{aligned} \quad (1)$$

Here, σ is the random variable to describe the target's coordinate. And $S(\mathbb{Y}_t,\sigma)$ is the distance between $\mathbb{Y}_t$ and σ.

Next, TinyG calculates the information entropy of the probability distribution. In information theory, the information entropy of a random variable is the average level of uncertainty inherent to the variable's possible outcomes. The higher the entropy, the flatter the probability distribution becomes, making it more difficult for us to estimate the location. The entropy

$H(\mathcal{M}_t)$ to describe the uncertainty at the end of round t can be expressed as:

$$H(\mathcal{M}_t) = -\sum_{\forall k} P(\mathcal{C}_k|\mathcal{M}_t) \cdot \log_2 P(\mathcal{C}_k|\mathcal{M}_t) \tag{2}$$

D. Phase 2: Predicting Entropy Reduction

As more measurements are conducted, our estimation of the location becomes more precise. In the first few rounds, there is significant uncertainty regarding the location, and TinyG aims to select a prober that can reduce the entropy more. By leveraging the delay-distance model, TinyG predicts the expected reduction in entropy for different probers in the next round. The algorithm for predicting entropy reduction is summarized and described in Algorithm 1. TinyG iterates over each city where the target may be located and predicts the entropy reduction separately, assuming the target is in that city. Finally, TinyG outputs the weighted mean of the entropy reduction, using the probability of each city as the weight. To reduce computation, TinyG selectively processes probers in the same city according to AS relationship.

To provide a detailed explanation of this process, we denote the expected entropy reduction of using $\mathbb{V}_i$ in round $t+1$ as $\mathcal{D}_{i,t+1}$. Assuming the target is located in $\mathcal{C}_k$, TinyG first obtains the estimated delay $\mathcal{T}^*_{t+1}$ based on the given AS relation $\mathcal{A}^*_{t+1}$ and the distance between $\mathbb{V}_i$ and $\mathcal{C}_k$. Next, TinyG **virtually** completes a measurement result $(\mathbb{V}_i, \mathcal{A}^*_{t+1}, \mathcal{T}^*_{t+1})$ and combines it with $\mathcal{M}_t$ to form $\mathcal{M}^*_{t+1}$. The predicted entropy reduction Δh^* of using $\mathbb{V}_i$ in round $t+1$, assuming the target is in $\mathcal{C}_k$, can be expressed as $H(\mathcal{M}_t) - H(\mathcal{M}^*_{t+1})$. Then, TinyG adds $P(\mathcal{C}_k|\mathcal{M}_t) \cdot \Delta h^*$ to $\mathcal{D}_{i,t+1}$, since the probability of the target being in $\mathcal{C}_k$ at the end of round t is $P(\mathcal{C}_k|\mathcal{M}_t)$. This process is repeated for each city, and the final output is the expected entropy reduction when using $\mathbb{V}_i$ in round $t+1$.

Algorithm 1 Predict Entropy Reduction

```
Input: $\mathbb{V}_i$; $\mathcal{A}^*_{t+1}$; $\mathcal{M}_t$; $\mathcal{C}_k(\forall k)$;
Output: $\mathcal{D}_{i,t+1}$
1: $\mathcal{D}_{i,t+1} \leftarrow 0$
2: for $k$ do
3:     $\mathcal{T}^*_{t+1} \leftarrow S2T(\mathcal{A}^*_{t+1}, S(\mathbb{V}_i, \mathcal{C}_k))$
4:     $\mathcal{M}^*_{t+1} \leftarrow \mathcal{M}_i \cup \{(\mathbb{V}_i, \mathcal{A}^*_{t+1}, \mathcal{T}^*_{t+1})\}$
5:     $\Delta h^* \leftarrow H(\mathcal{M}_t) - H(\mathcal{M}^*_{t+1})$
6:     $\mathcal{D}_{i,t+1} \leftarrow \mathcal{D}_{i,t+1} + P(\mathcal{C}_k|\mathcal{M}_i) \cdot \Delta h^*$
7: end for
8: return $\mathcal{D}_{i,t+1}$
```

E. Phase 3: Scoring Probers

When the entropy becomes sufficiently low, the probability differences between cities become more significant. In such cases, TinyG prefers probers from these high-probability cities. Rather than setting a fixed entropy threshold for switching strategies, TinyG utilizes a dynamic scoring mechanism. TinyG scores probers based on both the prediction of entropy reduction and the probability distribution. Let $\mathcal{P}_{i,t}$ denote the probability of the target being in the same city as $\mathbb{V}_i$ at the end of round t. Before scoring, TinyG shifts $\mathcal{D}_{i,t+1}$ to ensure that all of them are greater than or equal to 0. TinyG combines $\mathcal{D}_{i,t+1}$ with $\mathcal{P}_{i,t}$ and obtains the score $\mathcal{Z}_{i,t+1}$ as follows:

$$\mathcal{Z}_{i,t+1} = \mathcal{D}^{5/(t+1)}_{i,t+1} \cdot \mathcal{P}^{(t+1)/5}_{i,t} \tag{3}$$

TinyG selects the prober with the highest score for the next round. In the initial rounds, where the entropy is high and the probabilities of different cities are similar, the score is primarily influenced by $\mathcal{D}_{i,t+1}$. As the entropy decreases and the probability differences between cities become more pronounced, $\mathcal{P}_{i,t}$ plays a more significant role in the score. Additionally, we address the scenario where multiple probers achieve the same score. Considering that the presence at facilities and Internet Exchange Points (IXPs) reveals the potential distribution of the target, TinyG gives priority to probers in ASes present at the same facilities or IXPs as target's AS. The information on IXPs and facilities is obtained from PeeringDB [36].

IV. Evaluation

In this section, we first validate the assumption that geolocation accuracy is primarily determined by the minimum delay and show that the 2 ms delay can be used as the threshold for credible geolocation results. Then, we conduct experiments to evaluate how TinyG performs in reducing the ANP needed to obtain credible results. In addition, we measure the computational time of TinyG and demonstrate its acceptability.

A. Experimental Settings

Ground Truth. We create a hybrid ground truth dataset comprising 565 IP addresses belonging to M-Lab Pods [37] and 565 IP addresses of router interfaces from the CAIDA's Macroscopic Internet Topology Data Kit [38]. M-Lab is operated by Google and hosts servers worldwide. These servers are labelled with their respective cities and coordinates. The IP addresses of routers are obtained from the top-tier ASes of the global transit hierarchy, which play a vital role in the Internet. We extract the interface IP addresses from the routers in the top ASes according to CAIDA's ASRank [39]. We perform DNS lookups to retrieve their DNS hostnames. Through a combination of hostname-based geolocation methods [22, 26] and our manual verification, we obtain 565 routers' IP addresses with their corresponding city-level locations. While we acknowledge that the locations indicated by the DNS hostnames may not be entirely accurate, Zhang *et al.* [40] have shown that DNS misnamings occur in only a small fraction (0.5%). Therefore, it is reasonable to utilize them as a part of the ground truth.

Probers. We utilize LG probers located worldwide to conduct measurements. These probers offer users the ability to execute common measurement commands like `ping` and `traceroute`. Zhuang et al. [18] have compiled a comprehensive list of automatable LG probers, saving us the effort of searching for them. The web pages of these probers provide location labels, which are crucial for accurate IP geolocation. However, some prober locations may be inaccurately labeled due to operator negligence. Previous studies [12, 15, 29, 32] did not specifically address this issue. In our research, we take measures to identify and exclude probers with erroneous location labels to mitigate significant errors. To identify inaccurate prober locations, we perform delay measurements between randomly sampled pairs of probers. For each pair (a, b), we verify if they satisfy the *maxspeed-constraint*. If a pair fails to meet this constraint, it indicates that at least one prober's location in the pair is inaccurate. We keep track of the number of unsatisfied pairs associated with each prober and systematically remove probers in descending order based on the count of its unsatisfied pairs. By applying this approach, we successfully obtain a final set of 1,342 probers.

Baselines. To evaluate the performance of TinyG, we compare it with Hu *et al.* [15], whose details are introduced in Section II-B. To illstruate the effectiveness of TinyG, we define two multi-round prober selection algorithms, including MR-C and MR-P.

- **Hu *et al.* [15]**. Due to the frequency limits imposed by Looking Glass, we cannot directly use all probers to probe the representatives. Instead, we select one IP address as the representative from each prefix and use 10 probers geographically closest to the representatives according to the ground truth.
- **MR-C** (Multi-Round + *maxspeed-constraint*). At the end of each round, MR-C constructs feasible regoin based on the measurement results of previous rounds and randomly selects a prober located in the region for the next round.
- **MR-P** (Multi-Round + Probability). At the end of each round, MR-P calculates the probability distribution of the target. Then, MR-P randomly selects a prober located in the city with the highest probability for the next round. If there is no prober in the city, MR-P moves on to the next city.

Methodology and Metrics. To validate whether 2 ms can be used as the threshold for credible geolocation results, we feed the measurement results to various active IP geolocation methods and group the geolocation results based on their corresponding minimum delays. These results are then compared against the ground truth. Two widely used metrics, including city-level accuracy (ACC) and median distance error (MED), are used. In this paper, we consider geolocation results within a 40 km radius of the ground truth as accurate at the city level, a threshold previously utilized in various geolocation studies [9, 10, 14]. By comparing the ACC and MED of different delay groups, we demonstrate that geolocation accuracy primarily depends on the minimum delay and using a 2 ms delay as the threshold for credible geolocation results is justified.

To evaluate the effectiveness of TinyG in reducing the ANP needed for obtaining credible results, we compare TinyG with other prober selection algorithms. We combine two key metrics for evaluation: the true positive rate (TPR) and the ANP. TPR represents the fraction of IP addresses for which the algorithm successfully finds a prober within 2 ms compared to that of using all probers.

Hyperparameter Settings. Based on our observation, the TPR of TinyG remains nearly unchanged after $\mathcal{R}$ exceeds 20. Thus, we conduct experiments with a range of $\mathcal{R}$ from 1 to 20.

B. Relationship between Delay and Geolocation Accuracy

To validate the reasonability of using a 2 ms threshold for credible results and whether the geolocation accuracy is determined by the minimum delay, we feed the measurement results to active geolocation methods, including SP [29], CBG [12]. Then, we set a series of delay intervals and divide the geolocation results based on the associated minimum delay. The MED and ACC corresponding to different delay intervals are presented in Fig. 4.

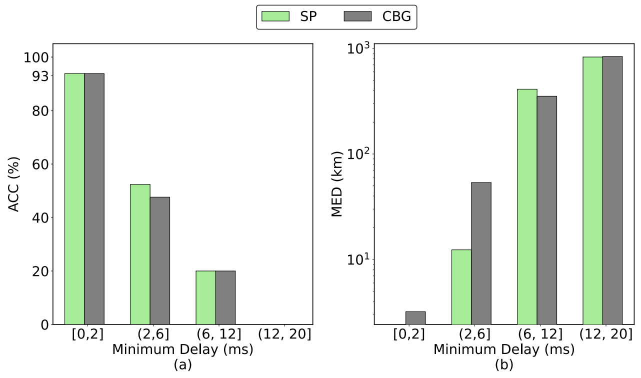

Fig. 4: (a) The ACC of different delay intervals. (b) The MED of different delay intervals.

When the minimum delay is within 2 ms, the ACC of the three IP geolocation methods is very close, and the ACC of SP and CBG is approximately 93%. However, as the minimum delay exceeds 2 ms, the ACC sharply declines. In the delay range of 2 ms to 6 ms, all of them exhibit an ACC below 60%.

The results show that the differences in ACC and MED caused by using various active geolocation methods are very small compared to the impact of varying minimum delays. Therefore, the minimum delay serves as the decisive factor for geolocation accuracy. Moreover, the results confirm that 2 ms is an important indicator for accurate geolocation at city-level, which aligns with previous research [14]. Therefore, 2 ms is the justified threshold for credible geolocation results.

Furthermore, we evaluate the ACC and MED of the most widely used IP geolocation database, MaxMind [8], on our ground truth dataset. The ACC of the commercial database of MaxMind is only 17%, which is 76% lower than our credible results. Therefore, when the minimum delay is within 2 ms, we recommend using the results obtained from active geolocation methods rather than querying popular geolocation databases.

C. Effectiveness in Reducing the ANP

To evaluate the effectiveness of TinyG in reducing the ANP needed to obtain credible results, we conduct a comparative analysis of the TPR and the ANP of different prober selection algorithms. Considering the *maxspeed-constraint* and the error of probers' coordinates, probers within 400 km from the targets are selected to investigate the coverage of credible results when using all probers. When using all probers, the coverage of credible results reaches 79.3%.

Fig. 5 displays the TPR of each algorithm under different $\mathcal{R}$ values. It is important to note that the parameter $\mathcal{R}$ is relevant **only** to multi-round algorithms, including TinyG, MR-P, and MR-C. Within the $\mathcal{R}$ range of 0 to 10, TinyG demonstrates a

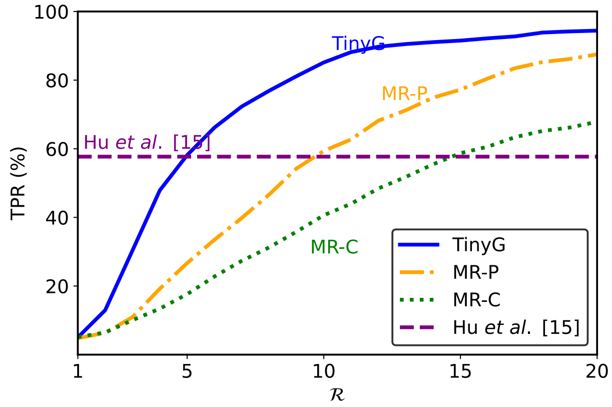

Fig. 5: The TPR of each prober selection algorithm under different $\mathcal{R}$.

rapid increase in TPR. When $\mathcal{R}$ is set to 20, TinyG achieves a high TPR of 94.4%, representing only 6% reduction compared to using all probers. In contrast, both MR-P and MR-C exhibit significantly lower TPR values. At $\mathcal{R}$ equals 20, their respective TPR values are only 87.5% and 67.9%. Furthermore, the TPR of [15] is observed to be only 57.7%. Then we compare the ANP values of different algorithms. It should be noted that for multi-round prober selection algorithms, the ANP is typically lower than $\mathcal{R}$ since they stop probing when the measured delay is within the 2 ms threshold. When $\mathcal{R}$ is set to 20, the ANP values of each algorithm are summarized in TABLE I. Notably, TinyG requires a remarkably low ANP of only 6.7. Overall, TinyG achieves the highest TPR with the minimum ANP, demonstrating its effectiveness in reducing the ANP needed to obtain credible results.

TABLE I: The TPR and the ANP of each algorithm when $\mathcal{R}$ is set to 20.

Algorithm	TinyG	MR-P	MR-C	Hu *et al.* [15]
TPR (%)	**94.4**	87.5	67.9	57.7
ANP	**6.7**	9.7	12.2	10*

* The probers for probing the representatives are not included.

Additionally, we analyze the poor performance of [15], which only achieves a low TPR of 57.7%. As discussed in Section II, [15] faces challenges when the IP addresses within a prefix are not geographically clustered. For instance, when two routers from different cities, countries, or continents are interconnected, the IP addresses on the link belong to the same /24 prefix IP, resulting in the prefix spanning very different areas. Fig. 6 displays the TPR of each algorithm on different

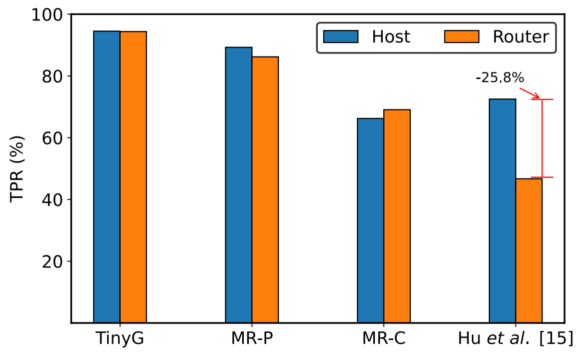

Fig. 6: The TPR of each algorithm on different device types (host and router) when $\mathcal{R}$ is set to 20.

subsets of the ground truth, including hosts and routers. Unlike other algorithms, the TPR on routers of [15] is significantly lower than on hosts. Furthermore, we examine the /24 prefix associated with each IP address of the routers and select prefixes contain IP addresses from at least 2 different cities, which account for 72% of IP addresses in the ground truth. Over 35% of the IP addresses in the ground truth are situated beyond a 400 km radius from their respective representatives. Consequently, the wide geographical span of these prefixes leads to the poor performance of [15], which is not illustrated in previous studies. Although our implementation is not entirely strict, the results can still reflect the poor performance of [15] in geolocating routers.

D. Computational Time

We further measure the computational time of TinyG because this is a crucial factor in determining whether it is practical to use in the real world. We deploy TinyG on a cloud server located in Beijing, China. In addition, we collect the HTTP response time of of each Looking Glass site for comparison. Conducting measurements using a public prober involves 3 remote interactions: (1) server→website; (2) website→prober; (3) prober→target. Thus, the response time is longer than that of common websites. The CDFs of these two types of time per round during our measurements is shown in Fig. 7(a). The average computational time is far below the HTTP response time, which is only 0.09 s. Therefore, the computational time caused by TinyG is acceptable.

V. Limitations

TinyG has two limitations: i) As TinyG selects different probers for different targets, it cannot be directly applied to delay vector-based methods such as Geoping [41], which use a fixed set of probers to probe each target, encoding the

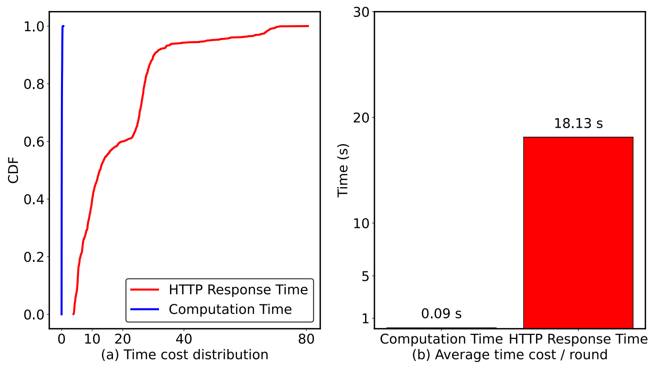

Fig. 7: Compared to the time cost of waiting HTTP responses of probers, the computational time of TinyG is negligible.

delays between different probers and the target as a fixed-length vector. The target is then geolocated through similarity comparisons or other techniques based on these vectors. ii) TinyG cannot be used for geolocating anycast IP addresses [42]. In anycast, multiple network nodes are associated with a single IP address. These nodes can be distributed across various locations. Different probers may receive responses from nodes at different locations.

VI. Conclusion and Future Work

In this paper, we propose TinyG, a multi-round prober selection algorithm designed to reduce the ANP in IP geolocation. TinyG divides the probing process into multiple rounds and utilizes the measurement results from previous rounds to guide prober selection in subsequent rounds. In each round, TinyG calculates the probability distribution of the target based on the measurement results of previous rounds and selects the prober for the next round based on entropy reduction prediction, AS relationship and other information. By feeding the measurement results to various active geolocation methods, including SP and CBG, we validate that geolocation accuracy is mainly determined by the associated minimum delay of the target, and 2 ms can be used as the threshold for achieving credible results. Experimental results demonstrate that TinyG outperforms other multi-round algorithms, improving the TPR while requiring smaller ANP. Compared to using all available probers, TinyG achieves a very small ANP of 6.7, with only a 6% decrease in the coverage of credible results. Overall, TinyG offers a practical solution for individuals and organizations seeking accurate and cost-efficient IP geolocation. Future work will focus on conducting large-scale measurements guided by TinyG to obtain more credible geolocation results, thereby providing a more comprehensive and accurate view of the Internet for the research community.

Acknowledgment

This work was supported in part by the Natural Science Foundation of China under Grant 62072269 and in part by the National Key Research and Development Program of China under Grant 2020YFE0200500.

References

[1] P. Callejo, M. Gramaglia *et al.*, "A deep dive into the accuracy of ip geolocation databases and its impact on online advertising," *IEEE Transactions on Mobile Computing*, 2022.
[2] M. Cozar, D. Rodriguez, J. M. Del Alamo, and D. Guaman, "Reliability of ip geolocation services for assessing the compliance of international data transfers," in *2022 IEEE European Symposium on Security and Privacy Workshops (EuroSPW)*, 2022.
[3] C. Iordanou, G. Smaragdakis, I. Poese, and N. Laoutaris, "Tracing cross border web tracking," in *Proceedings of the Internet Measurement Conference 2018*, ser. IMC '18, 2018, p. 329–342.
[4] M. Syamkumar, R. Durairajan, and P. Barford, "Bigfoot: A geo-based visualization methodology for detecting bgp threats," in *2016 IEEE Symposium on Visualization for Cyber Security (VizSec)*, 2016.
[5] R. Mahajan, M. Zhang, L. Poole, and V. Pai, "Uncovering performance differences among backbone isps with netdiff," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'08. USA: USENIX Association, 2008, p. 205–218.
[6] V. Giotsas, G. Smaragdakis, B. Huffaker, M. Luckie, and k. claffy, "Mapping peering interconnections to a facility," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '15. New York, NY, USA: Association for Computing Machinery, 2015.
[7] R. Padmanabhan, A. Schulman, D. Levin, and N. Spring, "Residential links under the weather," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. Association for Computing Machinery, 2019, p. 145–158.
[8] "MaxMind," https://www.maxmind.com/, [Online]. Available.
[9] M. Gharaibeh, A. Shah, B. Huffaker, H. Zhang, R. Ensafi, and C. Papadopoulos, "A look at router geolocation in public and commercial databases," ser. IMC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 463–469.
[10] Y. Shavitt and N. Zilberman, "A geolocation databases study," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 10, pp. 2044–2056, 2011.
[11] B. Huffaker, M. Fomenkov *et al.*, "Geocompare: a comparison of public and commercial geolocation databases - technical report," 2011.
[12] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint-based geolocation of internet hosts," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 288–293.
[13] B. Trammell and M. Kühlewind, "Revisiting the privacy implications of two-way internet latency data," in *Passive and Active Measurement*, R. Beverly, G. Smaragdakis, and A. Feldmann, Eds. Cham: Springer International Publishing, 2018, pp. 73–84.
[14] B. Du, M. Candela, B. Huffaker, A. C. Snoeren, and k. claffy, "Ripe ipmap active geolocation: Mechanism and performance evaluation," *SIGCOMM Comput. Commun. Rev.*, 2020.
[15] Z. Hu, J. Heidemann, and Y. Pradkin, "Towards geolocation of millions of ip addresses," ser. IMC '12. New York, NY, USA: Association for Computing Machinery, 2012.
[16] V. Giotsas, A. Dhamdhere, and K. Claffy, "Periscope: Unifying looking glass querying," 03 2016.
[17] "RIPE Atlas," https://atlas.ripe.net/, [Online]. Available.
[18] S. Zhuang, J. H. Wang, J. Wang, Z. Pan, T. Wu, F. Li, and Z. Zhang, "Discovering obscure looking glass sites on the web to facilitate internet measurement research," ser. CoNEXT '21.

Association for Computing Machinery, 2021.
[19] J. Udhayan and R. Anitha, "Demystifying and rate limiting icmp hosted dos/ddos flooding attacks with attack productivity analysis," in *2009 IEEE International Advance Computing Conference*, 2009.
[20] "Hurricane Electric's Network Looking Glass," https://lg.he.net/, [Online]. Available.
[21] D. Moore, "Where in the world is netgeo.caida.org?" in *Internet Society Conference*, 2000.
[22] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '02. New York, NY, USA: Association for Computing Machinery, 2002, p. 133–145.
[23] B. Huffaker, M. Fomenkov, and k. claffy, "Drop: Dns-based router positioning," *SIGCOMM Comput. Commun. Rev.*, 2014.
[24] Q. Scheitle, O. Gasser, P. Sattler, and G. Carle, "Hloc: Hints-based geolocation leveraging multiple measurement frameworks," in *2017 Network Traffic Measurement and Analysis Conference (TMA)*, 2017, pp. 1–9.
[25] "GeoNames," http://www.geonames.org/, [Online]. Available.
[26] M. Luckie, B. Huffaker, A. Marder, Z. Bischof, M. Fletcher, and K. Claffy, "Learning to extract geographic information from internet router hostnames," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT '21, 2021.
[27] C. Guo, Y. Liu, W. Shen, H. J. Wang, Q. Yu, and Y. Zhang, "Mining the web and the internet for accurate ip address geolocations," in *IEEE INFOCOM 2009*, 2009.
[28] Q. Li, Z. Wang, D. Tan, J. Song, H. Wang, L. Sun, and J. Liu, "Geocam: An ip-based geolocation service through fine-grained and stable webcam landmarks," *IEEE/ACM Transactions on Networking*, 2021.
[29] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe, "Towards ip geolocation using delay and topology measurements," ser. IMC '06. Association for Computing Machinery, 2006.
[30] R. Percacci and A. Vespignani, "Scale-free behavior of the internet global performance," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 32, pp. 411–414, 2003.
[31] G. Martinez, J. A. Hernandez, P. Reviriego, and P. Reinheimer, "Round trip time (rtt) delay in the internet: Analysis and trends," *IEEE Network*, pp. 1–6, 2023.
[32] S. Laki, P. Mátray, P. Hága, T. Sebők, I. Csabai, and G. Vattay, "Spotter: A model based active geolocation service," in *2011 Proceedings IEEE INFOCOM*, 2011, pp. 3173–3181.
[33] Y. Tian, R. Dey, Y. Liu, and K. W. Ross, "Topology mapping and geolocating for china's internet," *IEEE Transactions on Parallel and Distributed Systems*, 2013.
[34] O. Dan, V. Parikh, and B. D. Davison, "Ip geolocation using traceroute location propagation and ip range location interpolation," in *Companion Proceedings of the Web Conference 2021*, ser. WWW '21, New York, NY, USA, 2021.
[35] I. Youn, B. L. Mark, and D. Richards, "Statistical geolocation of internet hosts," in *2009 Proceedings of 18th International Conference on Computer Communications and Networks*.
[36] "PeeringDB," https://www.peeringdb.com/, [Online]. Available.
[37] "Measurement Lab," https://www.measurementlab.net/, [Online]. Available.
[38] "The CAIDA Macroscopic Internet Topology Data Kit - 2022," https://www.caida.org/catalog/datasets/internet-topology-data-kit/, [Online]. Available.
[39] "AS Rank," https://asrank.caida.org/, [Online]. Available.
[40] M. Zhang, Y. Ruan, V. Pai, and J. Rexford, "How dns misnaming distorts internet topology mapping," in *Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference*, ser. ATEC '06. USENIX Association, 2006.
[41] V. N. Padmanabhan and L. Subramanian, "An investigation of geographic mapping techniques for internet hosts," in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, 2001, pp. 173–185.
[42] D. Cicalese, D. Z. Joumblatt, D. Rossi, M.-O. Buob, J. Augé, and T. Friedman, "Latency-based anycast geolocation: Algorithms, software, and data sets," *IEEE Journal on Selected Areas in Communications*, 2016.