

# Fast Discovery of Live Internet Address Prefixes

Amgad Zeitoun Sugih Jamin

*Department of Electrical Engineering and Computer Science,  
The University of Michigan, Ann Arbor  
{azeitoun, jamin}@eecs.umich.edu*

**Abstract**—Several newly proposed services for the Internet require working with Internet hosts grouped into address prefixes (APs). Examples of such services include IP2Geo, IDMaps, network topology discovery, and Web cache and proxy placement. These services generally require reaching a host within an AP and collecting some measurements on that AP. We call the process of identifying a reachable host within an AP, *AP discovery*. According to BGP (Border Gateway Protocol) routing tables, there are presently more than 100,000 APs in use on the Internet. This poses a scalability bottleneck to services relying on AP discovery. Trying to identify reachable hosts by scanning all addresses within an AP is not only a slow process, but may also trigger false security alarms. We present an automated AP discovery algorithm that incorporates a set of heuristics for fast AP discovery, including *Random Selection, Neighbor Selection, and Conditional IP Selection*. By exploiting IP address assignment pattern, and targeting probe packets to a small subset of IP addresses, we can determine the reachability of the majority of APs in about half an hour, instead of a few days.

## I. INTRODUCTION

SEVERAL new services that take advantage of spatial locality of Internet hosts have been introduced in the past couple of years. For example, to help vendors target their advertisements, the IP2Geo family of tools [3] maps CIDR (Classless Inter-Domain Routing [1]) blocks to their respective geographic regions. Web client clustering heuristics proposed in [5] identify topologically co-located clients to help in the placement of Web caches, proxies, and mirrors. IDMaps [4] estimates distances between regions of the Internet to help clients find a close by copy of their target services. Various Internet mapping and topology discovery projects, such as [14], [15], [11], rely on running traceroutes to random IP addresses covering vast regions of the Internet to reconstruct the Internet topology.<sup>1</sup> To discover “topologically meaningful clustering<sup>2</sup>” of hosts on the Internet, such services usually start with the CIDR routing blocks advertised by BGP (Border

Gateway Protocol) [2]. In [5], the authors propose a client clustering method whereby addresses are first mapped into BGP-advertised CIDR blocks. Finer clustering is achieved either by clients’ DNS (Domain Name System) naming convention or by performing traceroutes to the addresses and matching the last two hops of the resulting paths. We show in Section II-A that relying on BGP-advertised CIDR blocks *alone* may not provide a topologically meaningful clustering of hosts. We consider DNS naming convention too arbitrary to provide reliable clustering. While the use of traceroute does pinpoint exactly the topological clustering of clients, due to its hop-by-hop nature, tracing routes to a large number of addresses is a very time consuming process.

We propose an end-to-end method to discover whether an AP has a reachable host. Our goal and the focus of this paper is *only* to find a reachable host within an AP, we do not dwell on the distance metric itself. Different applications may require different measure of distance. For some, a hop count may be sufficient, for others round-trip time, bottleneck link bandwidth, loss rate, etc. would be more appropriate. We use round-trip time as a measure of distance in this paper *for illustrative purposes only*. The focus of the paper is on finding a reachable IP address in an AP, not distance measurement. Once a reachable IP address is found, further application-specific measurements can be made to refine the APs. Efficient methods to conduct these measurements are outside the scope of this paper.<sup>3</sup>

One simple way to discover a reachable host within an AP is to pick IP (Internet Protocol) addresses in increasing order and, for each selected IP, send an ICMP echo request to it and wait for the reply. The discovery process will time out waiting for the reply if the selected IP address is unreachable. There are two immediately apparent problems with this approach. First, as pointed out by Braun *et al.* in [6], [7], only about 24% of valid IP address space is routable. Furthermore, based on historical usage patterns of telephone numbers, Huitema in [8] estimated that only about 0.3%–5.4% of IP address space will be occupied. Hence this method will not only waste

<sup>1</sup>A somewhat more detailed overview of these projects are provided in Section IV.

<sup>2</sup>By “topologically meaningful clustering” we mean clients within a cluster are equidistance from a measurement point on the Internet, where the distance metric is application specific.

<sup>3</sup>The point about this paper’s focusing *only* on finding a reachable IP within an AP and not on distance measurement will be made over and over again in the paper.

time probing unreachable IPs but also consume more network bandwidth than necessary. Second, due to the recent heightened sensitivity of network administrators to intrusion attempts, all probe packets are regarded with suspicion and hostility. Doing a simple walk down of the IP address space will raise a large number of false alarms. In our experience, adding randomization to the selection of IP addresses to probe does not perceptibly reduce the amount of false alarms.<sup>4</sup>

The discovery process can be made more intelligent by probing only those IP addresses that are registered with the DNS (Domain Name System). As we show in Section III-E, however, the set of DNS registered addresses and the set of reachable IP addresses are not proper subsets of either. Other approaches include extracting IP addresses from Web logs of popular Web servers [5], [3] and discovering Web server addresses using a Web crawler. Both of these approaches, however, cannot discover hosts within APs that do not have Web clients and servers.

As with most mapping and measurement efforts, we want our AP discovery process to map out as much of the Internet as possible in the shortest amount of time possible, with minimal perturbation to the system. Our approach exploits different IP address assignment characteristics observed through empirical experiments on the Internet. As described in Section III, we target our probe packets to only a few IP addresses within an AP. Our heuristics do not completely eliminate probing to unreachable IP addresses, nor do they eliminate falsely setting off network intrusion detection alarms. Nevertheless, we hope that by presenting them in this paper we can elicit better ideas from the community.

The rest of the paper is structured as follows. In Section II, we define terms used throughout the paper and present experiments and observations that led to the formulation of different heuristics. We evaluate our AP discovery technique on the Internet in Section III. In Section IV we present some related work. Finally, we summarize our work in Section V.

## II. ADDRESS PREFIX DISCOVERY

### A. Address Prefix

An IP version 4 (IPv4) address can be decomposed into two parts: the network part and the host part. The network part identifies the network on which the host resides, while the host part identifies the particular host on that network (host ID). The network part is also called the address prefix. The address prefix of an IP address can be ob-

<sup>4</sup>We have only anecdotal evidence that the number of emails warning us to stop probing does not reduce perceptibly.

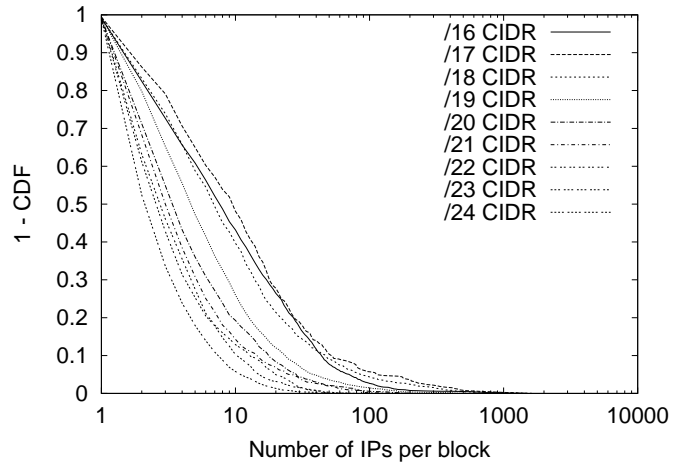


Fig. 1. Complementary cumulative distribution function of number of IPs per CIDR block.

tained by applying a bitmask to the address, with all the bits in the network part set to 1 and the other bits set to 0. This bitmask is called the *netmask* and is usually represented only by the number of 1's in the bitmask, i.e., the *prefix length*. For example,  $128.127/16$  means a netmask consisting of 16 bits of 1's in the higher order bits of the 32-bit IPv4 address. In BGP routing, consecutive address ranges that can be grouped together are aggregated into CIDR blocks. Aside from a simple hop count used in routing, CIDR blocks are usually not associated with measured distances. To measure the distance between a measurement point and a CIDR block, we first need to find a reachable host within that CIDR block. A host is reachable if it replies to probe packets sent by a measurement machine located somewhere on the Internet. For ease of exposition we differentiate between CIDR blocks and Address Prefixes (APs) by defining an AP to be a client cluster within a consecutive address range *equidistance from a measurement point on the Internet*. Thus, we define two types of AP: *passive* and *active*. A passive AP is an AP for which no reachable hosts has been discovered. Initially, all CIDR blocks extracted from BGP routing table are considered passive APs. If the discovery process could successfully reach a host within a passive AP, the AP becomes active (we interchangeably call an active AP a *live* AP).

Most services requiring clustering of Internet hosts start with the CIDR blocks obtained from BGP. If topological distance is important to an application, however, CIDR blocks by themselves do not provide a fine enough granularity of clustering. To show this, we collected 215,862 Web and FTP servers' addresses. These IP addresses map into 15,479 CIDR blocks. Clearly, the 215,862 IP addresses are not uniformly distributed across the 15,479

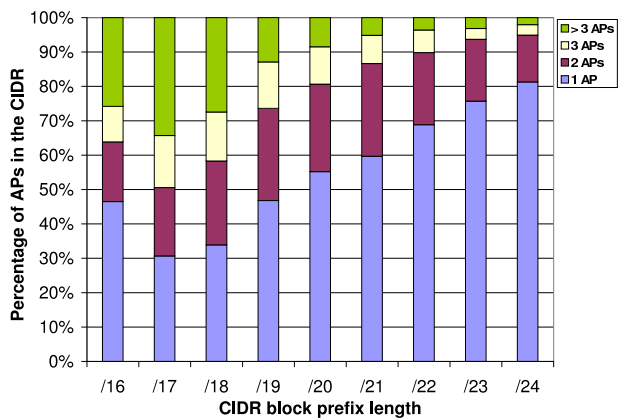


Fig. 2. The percentage of APs per CIDR block.

CIDR blocks. For each count of IP address(es) falling into one of these CIDR blocks, we add up the number of CIDR blocks having that count. Figure 1 shows the complementary cumulative distribution function (1-CDF) of the probability that a CIDR block has a given count of IP addresses ( $x$ -axis) fallen into it. For CIDR blocks with more than one IP address, we next measure the round-trip time to each address. Based on these measurements, we sub-divide each CIDR block into APs by round-trip time. IP addresses with round-trip times within 10% or 5 msec of each other are considered to belong to the same AP. Figure 2 shows that a substantial fraction of CIDR blocks, especially those with prefix lengths shorter than 24 bits, can be decomposed into multiple APs.<sup>5</sup>

Figure 3 shows the cumulative distribution function of the prefix lengths of these 15,479 CIDR blocks (the “CIDR of mapped IPs” curve). As can be seen from the figure, more than 70% of these 215,862 IP addresses fall into CIDR blocks with prefix lengths shorter than 24 bits. To compare this distribution against that of the Internet in general, we downloaded one BGP snapshot per month (from March 2000 to February 2001) from the National Laboratory for Advanced Network Research (NLNAR) Web site. The number of unique CIDR blocks grew from about 77,000 in March 2000 to nearly 106,000 in February 2001. The “All CIDR in BGP” curve in Figure 3 shows the cumulative distribution function of prefix lengths advertised by BGP in February 2001 (other snapshots, including those more recent than February 2001, show similar distribu-

<sup>5</sup>We caution that Figure 1 shows that for a large number of these CIDR blocks we have only 2 or 3 IPs. In general, the *only* claim we are making here is that if host clustering based on topological distance is important to an application, CIDR blocks do not provide a fine enough granularity of clustering. Each CIDR block can usually be decomposed into topologically distinct APs. While our data support the claim that there could be more than one AP per CIDR block, it *does not establish any quantitative claim on the exact number of APs per CIDR block.*

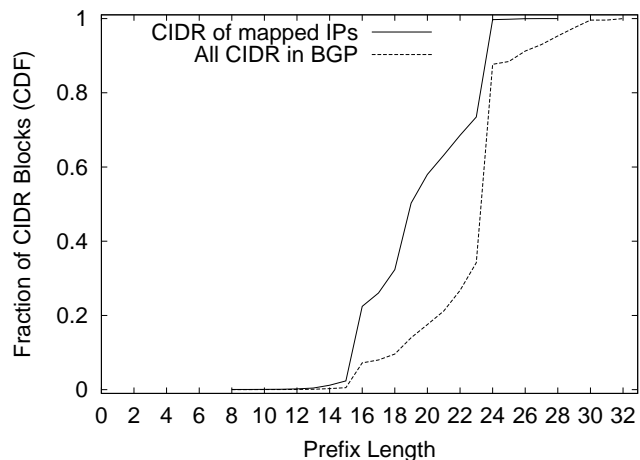


Fig. 3. Distribution of prefix length of CIDR blocks for the mapped 215,862 IPs and for all CIDR blocks advertised in a given BGP snapshot.

tion). From the figure, we can see that about 38% of the CIDR blocks advertised by BGP has prefix lengths shorter than 24 bits. The large fraction of CIDR blocks with prefix lengths shorter than 24 bits further emphasizes the need for client clustering methods that do not rely solely upon CIDR blocks obtained from BGP routing tables.

Although CIDR blocks advertised in BGP routing tables can have prefix lengths *longer* than 24 bits, there is a consensus among network administrators that such CIDR blocks may be filtered out by BGP routers to reduce routing table size [9]. The longest prefix length that *must not* be filtered out is /24. Following this consensus, we adopt /24 as the minimum AP size in our AP discovery process. We decompose CIDR blocks with shorter netmasks, i.e., less than 24-bit, into multiple /24 passive APs and apply the same heuristics described in Section II-B to each, in search of a reachable IP.

We next turn to the question: given a /24 CIDR block obtained from BGP (either advertised as a /24 CIDR block, or as part of a CIDR block with a shorter prefix), what is the most effective way to determine whether it has a reachable IP? Once a reachable IP has been found, the distance to it can be measured and the boundaries of an AP can be determined. We repeat our earlier statement that efficient methods to conduct these measurements are outside the scope of this paper. The focus of the paper is on finding *one* reachable IP within an AP, not distance measurement.

## B. Conditional IP Selection

A 24-bit CIDR block has potentially 254 reachable hosts—excluding host IDs<sup>6</sup> 0 and 255, which are used to

<sup>6</sup>“Host ID” and “IP address” are used interchangeably in this paper.

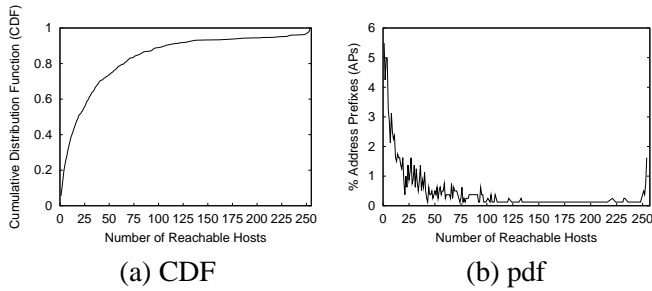


Fig. 4. Number of reachable hosts discovered in different active APs.

identify the network and broadcast addresses, respectively. Theoretically, we can probe host ID 255 or host ID 0 and wait for replies from all reachable hosts within that AP [10]. Due to security concerns, however, broadcast probes are summarily filtered out by every router on the Internet. For similar reasons, queries using Simple Network Management Protocol (SNMP) to obtain host ID information may be disabled or restricted on remote networks. The alternative, to systematically probe each host ID within that AP to discover a reachable one<sup>7</sup> is not only a slow process, but also has a high probability of triggering false security alarms at remote sites. Hence, in order to accelerate the discovery process, we send probes only to a small subset of all available host IDs within a passive AP. There is a trade-off between minimizing the number of host IDs within this subset, and maintaining a high reachability factor, i.e., probing cost vs. accuracy of AP discovery. In this subsection we formulate a set of methodologies that balances these trade-offs.

We first try to determine how many host IDs within one AP to test for reachability. Then we try to determine which host IDs to select. To these ends, we set up an experiment by randomly selecting 1,000 24-bit APs from IPv4 Class-C addresses.<sup>8</sup> We verify that these APs exist in all snapshots of BGP routing tables gathered. Recall that the main goal of AP discovery and the focus of our paper is to discover *one* reachable host and not *all* reachable hosts within an AP. In this experiment, however, we want to determine the optimal number of host IDs to test for reachability,

<sup>7</sup>For ease of experimentation, we use either the ICMP (“Echo Request” and “Echo Reply”) packets or TCP (“SYN” and “SYNACK/RST”) packets in our probing. Probe packets to reachable hosts return within a single round-trip time, while probe process to unreachable hosts times out after a fixed interval, usually on the order of a few seconds.

<sup>8</sup>We tried to expand our experiment by discovering another 1,000 24-bit APs from the range of the classical “Class-B” addresses and 650 APs from the range of the classical “Class-A” addresses. Unfortunately, we received a large number of complaints from the network administrators of these APs, which forced us to terminate our experiment. Nevertheless, our overall AP discovery technique selects APs from all available address ranges, as described in Section III-C.

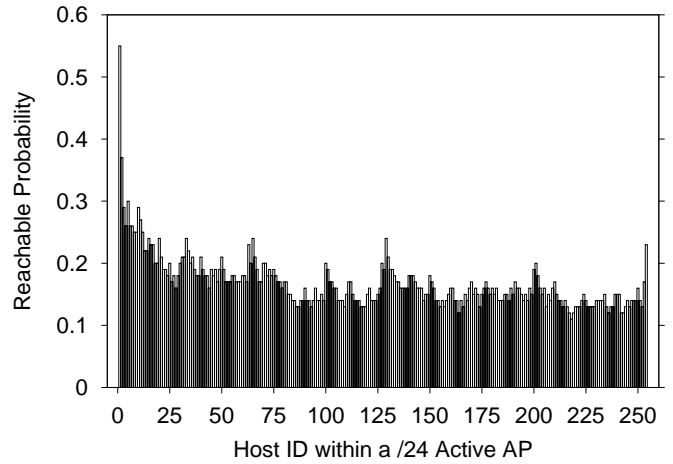


Fig. 5. The reachable probability of each host ID within different active APs.

along with their identities, hence in this experiment we need to discover *all* reachable hosts within an AP. Therefore, for each AP, we send probe packets to all 254 of its host IDs, spaced out at large intervals to reduce the chance of triggering false security alarms at remote sites. We test the reachability of a different host ID once every 2 hours, on average.

Of the 1,000 passive APs used in our experiment, 199 remained passive, i.e., failed to reply to any of our probes, this may happen due to network reachability problem at the remote site or due to packet filtering by routers or firewalls.<sup>9</sup> Of the other 801 active APs, the minimum number of hosts discovered is one, and the maximum is 254 (100% of total address space). The average number of reachable hosts within an active AP is about 43 (17% of the total address space). Figure 4 shows the Cumulative Distribution Function (CDF) and the probability density function (pdf) of the number of reachable hosts in these active APs. The figure shows that 80% of the APs have less than 65 reachable IP addresses out of 254 possible (25.6% utilization).

It is well known that host IDs within a 24-bit netmask address space are not equally likely to be reachable. Here we quantify the likelihood that a given host ID is reachable. Figure 5 shows the average frequency of reachability of each host ID within the 801 active APs found in our experiment. The  $x$ -axis represents the host IDs within a /24 AP, the  $y$ -axis represents the corresponding frequency of occurrence of each host ID in all 801 active APs.

<sup>9</sup>When a host ID is not reachable from our measurement host, it could also be caused by our measurement host itself suffering connectivity problems. To check the connectivity of our measurement host, our discovery process periodically confirms connectivity to a couple of well-connected Web sites; if connectivities to these Web sites fail, the discovery process is suspended until connectivity is restored.

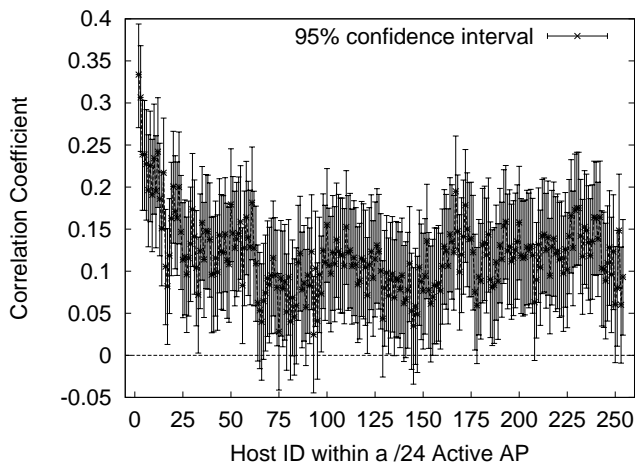


Fig. 6. Correlation coefficient between host ID 1 and other host IDs, with 95% confidence interval.

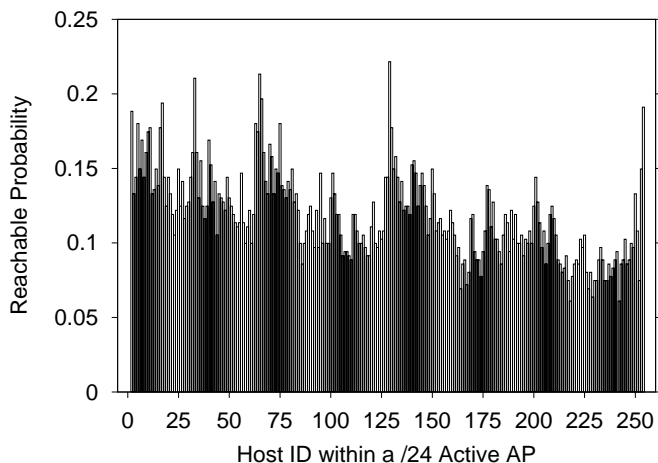


Fig. 7. The reachable probability of each host ID within different active APs, given that host ID 1 is unreachable.

For example, host ID 1 has the highest frequency of reachability, it replied to our probe packets in about 55% of the 801 active APs.

We only need to discover one reachable host within an AP to declare the AP active. We want a discovery process that picks a set of IP addresses that have a high probability of being reachable, to minimize the discovery time. Since host ID 1 has the highest probability of being reachable within an AP (Figure 5), our discovery process first sends probe packets to host ID 1. If host ID 1 does not reply, which host ID would be the best one to test next? Figure 5 shows that host ID 2 has the next highest probability of occurrence. If the frequency of occurrence of different host IDs in an AP is not independent, however, it may be likely that a probe to host ID 2 will also fail if the probe to host ID 1 has already failed. To determine whether host ID allocations are independent, we calculate the correlation coefficient ( $r$ ) between host ID 1 and host ID  $i$  ( $r_{1i}$ ),

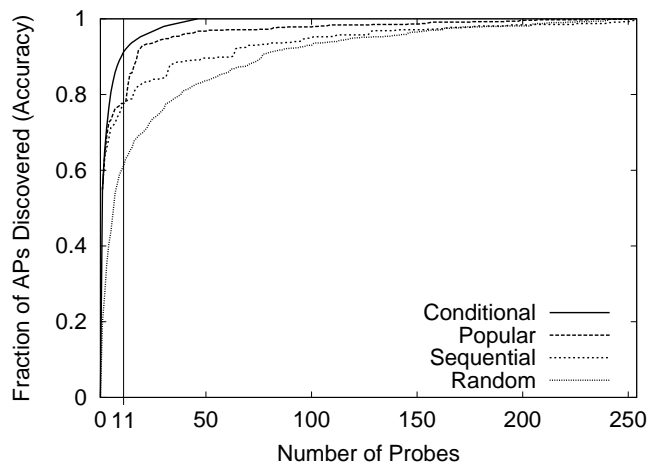


Fig. 8. Number of probes required vs. discovery accuracy.

$i \in \{2, 3, 4, \dots, 254\}$ . Figure 6 illustrates the correlation coefficient with the 95% confidence interval. Although the correlation coefficient values for different host IDs are less than 0.35, there is a statistically significant relationship between host ID 1 and the first 60 or so host IDs in the address range, i.e., the 95% confidence intervals do not fall to 0. This indicates that the allocation of host IDs in this region may be correlated with host ID 1.

By looking at the reachability of host IDs in active APs for which host ID 1 is unreachable (Figure 7), we find that the distribution of reachability has changed from Figure 5. Figure 7 shows that if host ID 1 is unreachable, the next most probably reachable host ID is 129. If we rank the host IDs by their frequency of occurrence, conditioned upon host ID 1 being unreachable, we get: 129, 65, 33, 66, 17, 254, 2. Thus, host ID 2 moved from the 2<sup>nd</sup> rank to the 7<sup>th</sup> rank. Host ID 129 being the next most probable to be reachable, given that host ID 1 is unreachable, indicates the popularity of subnetting 24-bit CIDR blocks into 25-bit CIDR blocks, where host ID 129 is the first host ID in the  $x.x.x.128/25$  address range.

Based upon the conditional probability of a host ID being reachable, given that all preceding host IDs are not reachable, we schedule our first 11 probes to the following host IDs (the numbers in the parentheses are the addresses' *conditional probability* of reachability): 1 (.55), 129 (.222), 254 (.18), 2 (.182), 64 (.176), 5 (.148), 33 (.148), 11 (.124), 17 (.111), 101 (.102), and 127 (.114). This means that if the measurement to host ID 129 also fails, host ID 254 will be the most probable one to be reachable, and so on. We call this the “Conditional” method of probing. Applying the “Conditional” probing to the 801 active APs we have discovered, we can determine the liveness of more than 91% of them using just these 11 host IDs. We say that “Conditional” probing has

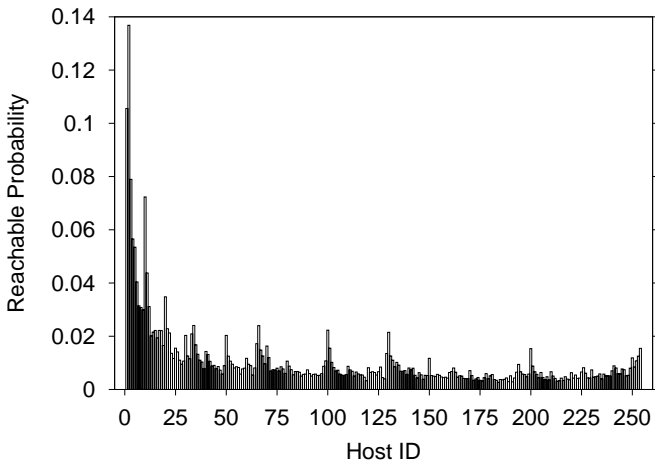


Fig. 9. Host IDs distribution within /24 CIDR blocks for Web and FTP servers.

an accuracy of 91% with 11 probes. In contrast, if the first 11 host IDs to be probed were picked randomly with uniform distribution (the “Random” method), the accuracy achieved is only 61%. Selecting the first 11 host IDs to probe by sequential scan (the “Sequential” method) or based on their popularity (i.e., unconditional probability, the “Popularity” method), the first 11 probes managed to determine the liveness of only about 77% of the 801 active APs. Figure 8 shows the accuracy achieved under the four probing methods for varying number of probes. Depending on the accuracy required of the AP discovery process, a service can choose the number of probes it sends per AP. For example, to achieve 100% accuracy under our “Conditional” method, i.e., discover *all* 801 active APs in our experiment, we need to probe a maximum of 46 host IDs (out of 254 possible) per AP.

After the first 11 host IDs, the conditional probability of the remaining host IDs are not significant enough to induce any meaningful ordering. Instead, we identify regions of host IDs with the highest probability of reachability. Of the 215,862 Web and FTP servers’ addresses we have used in Section II-A, 16,836 reside in 4,071 /24 CIDR blocks. Figure 9 shows where in the /24 address space these IPs reside. The figure shows obvious peaks at various regions of the address space.

### C. Automated AP Discovery Algorithm

So far we have studied how to select IPs within an AP to minimize the time needed to determine the liveness of an AP. In this subsection we develop an automated AP discovery process. We ask, given a list of /24 CIDR blocks obtained from BGP, how should we go about selecting the next one to probe to improve the probability of discovering

a live AP? Below we look at two heuristics: *Random Selection* and *Neighbor Selection*.

**Random Selection (RS):** *Randomly select a /24 passive AP for discovery from the set of all available passive APs. For passive APs with shorter netmasks, split them into several /24 passive APs.*

In order to optimize the passive AP selection heuristic, we study the relationship between neighboring APs. Each /24 address range has a left and right neighbors (except the first and last ranges in the IPv4 address space). For example, the range 128.127.10/24 has ranges 128.127.11/24 and 128.127.9/24 as neighbors. If an AP has been found active, what is the likelihood that its neighbors will also be active? To answer this question, we first extract the set of neighboring APs from the 1,000 APs used in Section II-B. This set contains 470 APs, 378 of which are determined to be active using our heuristics described in Section II-B. Next, we calculate the conditional probability that one or both neighbor(s) of  $AP_i$  is/are active given that  $AP_i$  is active. We find  $P[AP_{i\pm 1} = \text{active} | AP_i = \text{active}] = 89\%$ , very high. This leads us to the following refinement of the *Random Selection* heuristic.

**Neighbor Selection (NS) Optimization:** *For each active AP discovered, check its neighboring /24 APs for reachable hosts before trying another passive AP selected by RS.*

In other words, the NS optimization can be applied if some /24 active APs have been discovered, *and* some of the /24 neighboring APs of the active APs have not been tested for liveness. If the NS optimization can be applied, both /24 passive neighbors of the active AP are scheduled for measurement. Otherwise, the random selection process selects another /24 passive AP. The RS process may split passive APs with netmasks shorter than 24 bits into several /24 passive APs. A similar heuristic was used in the Mercator project [11].

One can draw an analogy between our AP selection method and the classic game of Battleship [12], the RS process first tries to guess the locations of active APs at random places on the Internet. Once an active AP is located, the NS optimization extends the selection coverage around this active AP. Given the high probability of discovering active APs in the neighborhood of other active APs (89%), the NS optimization can increase the AP discovery rate.<sup>10</sup>

<sup>10</sup>This observation can be further exploited by initially determining the liveness of an AP through some external means such as from the client lists of a Web server or from the results of conducting a Web crawl.

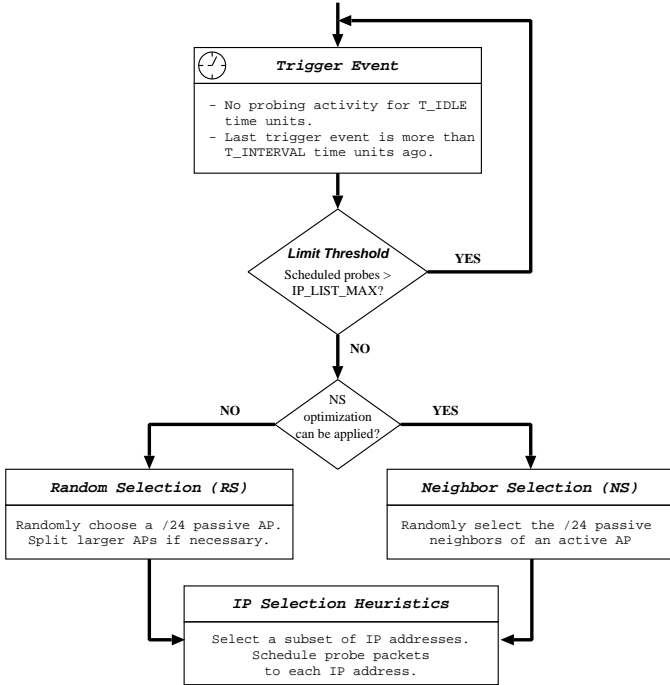


Fig. 10. AP discovery process.

In summary, our automated AP discovery process consists of two parts. Initially we select a set of APs to probe; then for each AP, we choose a number of IPs to check for reachability by scheduling probe packets to them at different times. The scheduled probes to different IPs in different APs are interleaved, i.e., the discovery process tries to determine the liveness of multiple APs in parallel. As our AP discovery process progresses, we continually replenish the set of APs to probe before the current list of scheduled probes is exhausted. After initial seeding of live APs, the choice of which passive AP to schedule for probing next is first done by using the neighbor selection (NS) process. If the NS process fails to select a passive AP, we use random selection (RS) to select a passive AP. Recall that for each AP to probe, we schedule a set of probe to different IP addresses within that AP.

To prevent our probe packets from overwhelming the network where measurement hosts are located, we limit the number of APs being concurrently probed. Probes destined to each AP are also spaced out at an interval that minimizes the probability of triggering false security alarms. Once a given number of probes are scheduled (IP\_LIST\_MAX), we say that we have reached a *limit* threshold and suspend the scheduling of any more probes. If all probes destined for an AP failed to elicit any response, we declare the AP passive. If, however, one of the probes elicits a response, we cancel the remaining probes destined for the same AP. Eventually, all the scheduled probes would have either been launched or canceled. If

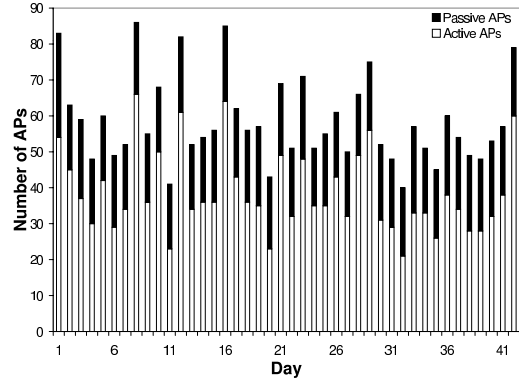


Fig. 11. The total number of passive APs tested per day with the fractions that became active or stayed passive.

no measurement packets have been sent for T\_IDLE time units we resume scheduling more APs for discovery. We call this a *trigger* event. We also generate a trigger event once every T\_INTERVAL since the last trigger event. Figure 10 summarizes our automated AP discovery process.

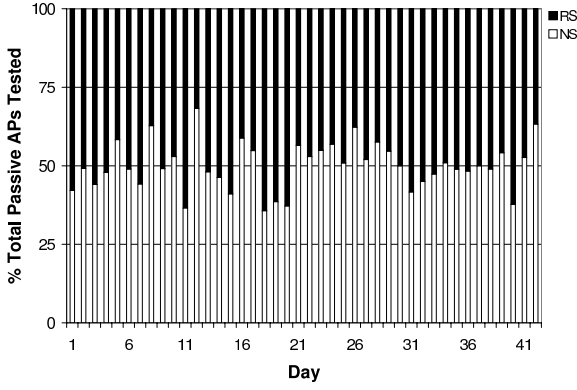
### III. AP DISCOVERY EVALUATION

To evaluate the effectiveness and generality of our AP discovery process, we obtained an initial set of passive APs (CIDR blocks) from a March 2001 BGP routing table snapshot. There are more than 115,000 passive APs with different prefix lengths in this set, covering address ranges spanning all of the three classical IP version 4 classes. We let our AP discovery process run for a period of 42 days on one measurement machine local to our site. To isolate the effectiveness of our heuristics, we did not use any external information, such as DNS records, Web server logs, or Web crawler results, to determine the liveness of an AP. Instead, we generate IP addresses to probe solely from our heuristics described in Section II. To study the effect of IP selection, we schedule 43 probes to different IP addresses within each AP selected. Given the accuracy of the first 11 probes when our “Conditional” probing method is used (see Section II-B), we concentrate on evaluating the discovery success rate of the first 11 probes.<sup>11</sup>

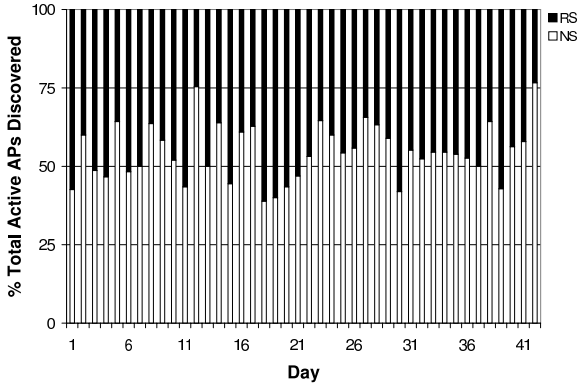
#### A. AP Discovery Rate

Given the intentionally slow pace of our AP discovery process, we only managed to examine 2,453 passive APs by the end of our 42-day experiment, 1,624 (66%) of these were found to be active. Some 44% of the active APs cover address ranges that form part of the classical “Class-A”

<sup>11</sup>For the experiment reported here, we set T\_IDLE to 1 minute and T\_INTERVAL to 30 minutes. These numbers did not trigger any false intrusion detection security alarms. To avoid rousing the ire of network administrators, we did not attempt to determine the smallest values that do not trigger false security alarms.



(a) Passive APs tested by the RS and NS processes.



(b) Active APs discovered by the RS and NS processes.

Fig. 12. The percentage of passive APs tested and discovered by RS selection and NS optimization per day.

range, 26% cover the “Class-B” range, and the remainder covers “Class-C” range. This results indicate that *despite operating only on /24 APs, our heuristics can discover APs from larger address blocks.*

We next look at the day to day progress of the AP discovery process. Figure 11 shows the total number of passive APs examined per day. The height of each bar in the figure represents the total number of passive APs examined for each of the 42 days, the white region of each bar represents the number of APs found to be active, while the black region represents those that remained passive. Figure 12(a) shows the percentage of examined APs initiated by the RS and NS processes, and Figure 12(b) shows these percentages for the APs that were found to be active. The number of APs examined each day depends largely on the number of active APs discovered. Recall that we send at most 43 probes to each AP in search of a reachable host. If a reachable host is found early in this process, the remaining probe packets destined for that AP are canceled and hence more APs can be examined. For example, the

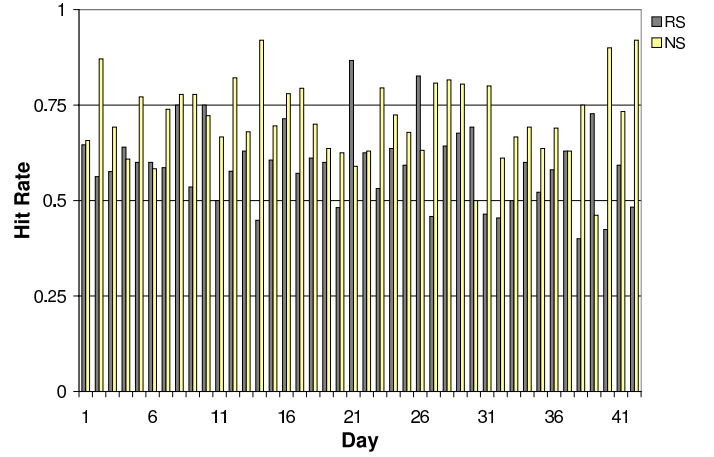


Fig. 13. The hit rate for APs examined by RS and NS processes.

number of passive APs examined at days 1, 8, 12, 16, and 42 is quite large compared to other days (see Figure 11). From Figure 12, we can see that on these days—except the first day where the start-up effect is more significant—the number of active APs discovered by NS optimization is larger than those by RS (more than 75% on some days). Due to the fact that active APs tend to co-locate, NS optimization discovers populated address spaces close to each other. The discovery time for the majority of active APs is less than half an hour (see Section III-D).

### B. Effectiveness of the AP Selection Heuristics

Of the total 2,453 APs examined, 1,203 (49%) were initiated by the RS process. Of the 1,624 active APs, 716 (44%) were initiated by the RS process, which give this process a 59.5% “hit rate” ( $\frac{716}{1,624}$ ). The NS process initiated the examination of 908 of the active APs, 56% of the total, giving it a hit rate of 72.6%. Turning to the daily statistics, Figure 13 shows the hit rate of APs examined by the RS and NS processes. The figure re-confirms our earlier finding that the probability of discovering an active AP in the neighborhood of existing active APs is high and further justify our use of the NS process.

### C. On the Generality of Our Results

In Section II-B, we noted that of the 801 active APs in our sample space of 1,000 APs, more than 91% of them can be determined to be active by the first 11 probes. Of the 1,624 active APs used in this section, only *one* was also present among the 801 found in Section II-B. *We emphasize that this latter set of active APs do not overlap significantly with the control set used in Section II-B.* Figure 14 shows the CDF of the number of probe packets required to discover an active AP, for all 1,624 active APs. *The figure shows that our conditional scheduling of probes can*

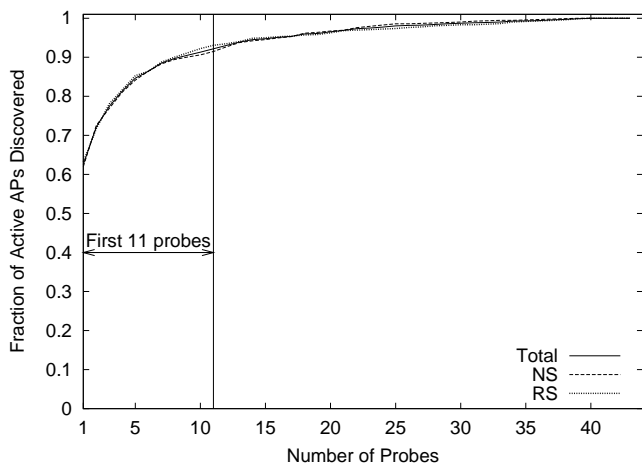


Fig. 14. Number of probes required to discover an active AP for RS and NS optimization.

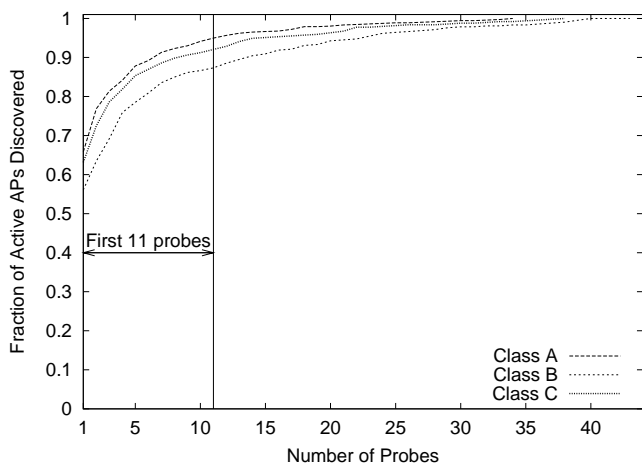


Fig. 15. Number of probes required to discover an active AP in different address classes.

be generalized to a larger population beyond our sample space in Section II-B. Using our “Conditional” probing method, more than 62% of the 1,624 active APs were discovered by our first probe packet to host ID 1, with the first 11 probes discovering more than 92% of the 1,624 active APs. This is true regardless of the methodology used for selecting the AP for examination (RS or NS).<sup>12</sup>

The conditional scheduling for probes was formulated Section II-B based on our observations made from /24 APs covering address ranges that are part of the classical Class-C address space. Figure 15 shows that applying the methodology to address ranges that are part of the classical Class-A and Class-B address spaces resulted in comparable performance.

<sup>12</sup>In this experiment we send a maximum of 43 probes per AP.

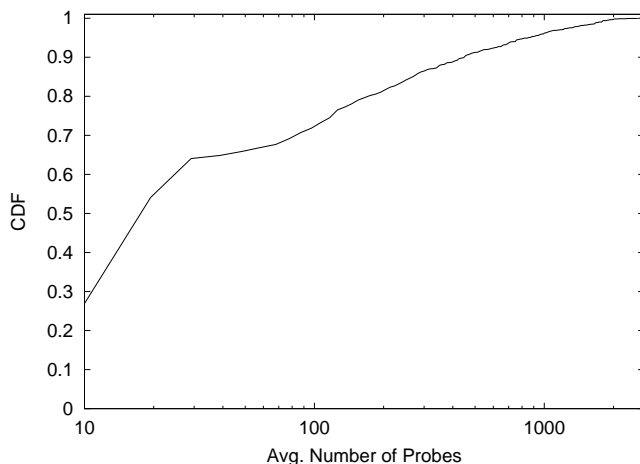


Fig. 16. The cumulative distribution function of the average number of probes, sent by the AP discovery process, needed to discover our 1,624 active APs set.

#### D. Discovery Time of Active APs

Aside from the selection of the AP to examine (RS and NS) and the choice of IP addresses within each AP to send probes to, the time it takes to discover an active AP depends largely on the rate at which the probes are sent. In this section we present statistics on time between subsequent AP discoveries in units of “number of probes.” Recall that we interleave sending probes to multiple APs concurrently. So the number of probes include those sent to APs that remained passive. Figure 16 shows the CDF for the discovery time of all 1,624 active APs in our experiment. Using our heuristics, about 28% of all APs are discovered with  $\leq 10$  probes. More than 50% of the active APs were discovered within the time to send 20 probes (in our case, this translates to about 12 minutes) and 80% of them with less than 200 probes (less than 2 hours).<sup>13</sup> One way to increase the AP discovery rate is to decrease the inter-packet gap of the probes. Doing so however, may increase the probability of triggering false security alarms. Alternatively, one can assign different regions of the IP address space to several measurement hosts, to further parallelize the AP discovery process.

#### E. Discussions

Domain Name System (DNS) stores information about different IP addresses and their corresponding canonical names (domain names) and provides name-to-address mapping service. We study whether IP addresses resolved by DNS have a high probability of being reachable, i.e.,

<sup>13</sup>The exponential increase in the region below 30 probes in Figure 16 reflects the discovery of APs with reachable host ID 1 (compare Figure 14).

can we base our IP selection heuristics to those IPs that are resolvable by DNS and hence further improve the AP discovery process. To that end, we conducted an experiment whereby we query the DNS for all IP addresses within the 1,000 passive APs used in Section II-B. According to DNS, 592 of the passive APs have at least one resolvable IP. Out of these 592, 541 were discovered by our probing exercise in Section II-B. The average number of IP addresses resolved per passive AP is 83. The important observation is that IP addresses resolved by DNS are not necessarily reachable by our probe packets. On the other hand, some of the IP addresses we found reachable are not resolvable by DNS. Only 26.7% of DNS resolvable IPs are reachable by our probe packets, whereas 61.7% of the IPs reachable by our probe packets are *not* resolvable by DNS. This implies that the majority of IP addresses can be reachable without having DNS records. Hence, our IP selection heuristic cannot optimize its IP selection strategy based on DNS resolution.

One limitation of our AP discovery process concerns hosts behind proxies or firewalls (such as in the case of AOL [3]). Probe packets directed to hosts behind a firewall could be dropped or replied to by the firewall. In the latter case, the AP discovery process will identify the firewall as the reachable host. In the former case however, our AP discovery process will fail. Unless there is a cooperating hosts behind the firewall/proxies that inform our AP discovery process of the network structure behind the firewall/proxies, we will not be able to map out that part of the network. This limitation applies to all of the mapping algorithms we are aware of.

Finally, our AP discovery techniques do not guarantee the reachability of discovered IP addresses at all times. A reachable host may crash or be turned off, rendering the host unreachable, or the network on which the host resides may become unreachable due to routing instability. Thus, services using our AP discovery technique should periodically check the reachability of discovered hosts, should that information be important to the service.

#### IV. RELATED WORK

There are a number of research projects in the area of Internet topology discovery. Internet mapping projects such as the ones presented in [14], [15], [11] attempt to reconstruct router-level topology of the Internet by tracing the routes to random IP addresses. The fidelity of the reconstructed topology partly relies upon the coverage of the selected destination addresses. Ideally, each topological regions of the Internet will be represented in the selected addresses. The mapping utility in [15] obtains CIDR blocks from BGP routing tables and sequen-

tially scans each CIDR block to look for a reachable host. The tool used in [11] picks IP addresses randomly from each address region. Our proposed “Conditional” probing method can be in such mapping tools.

IDMaps [4] is a distance estimation service that computes the distance between any two point on the Internet. To be scalable, instead of measuring and storing the distance to each individual host, IDMaps groups hosts into APs and measure and store only the distance to each AP. To look up the distance between two hosts, IDMaps first maps the hosts to their corresponding APs. A fast and efficient way to locate reachable hosts within different APs is thus crucial to IDMaps.

In [5], the authors used CIDR blocks extracted from BGP routing information to identify clients’ clusters to help in placement of Web proxies or mirrors. Client IP addresses were obtained from the Web logs of different Web servers. Each client is mapped to a CIDR block by longest prefix match. The authors introduced the cluster-based Internet topology modeling in [17]. In this model the Internet is viewed as a set of inter-connected clusters. Connectivity between clusters is determined by running `traceroute` to random IPs in each cluster.

IP2Geo [3] is a suite of techniques—*GeoTrack*, *GeoPing*, and *GeoCluster*—that map IP addresses to their geographic locations. The tools obtain IP addresses from Web logs and identify their geographic locations based on various techniques such as geographic clues in their DNS names. IP addresses are then mapped to their CIDR blocks to geographically locate each CIDR blocks. Our proposed methodologies can be used in conjunction with IP2Geo to find a reachable host in CIDR blocks for which no IP address is present in available Web logs.

The Skitter project [16] uses a process similar to `traceroute` to provide a router-level map of the Internet along with the RTT values for different paths. To do so, Skitter feeds a set of measurement machines distributed across the Internet with a list of destinations (DNS servers and Web logs). Apparently, our discovery technique enables Skitter to target its measurements to large portion of the addressable space on the Internet.

There have also been studies on IP address utilization on the Internet. For example, the authors of [6], [7] extract CIDR blocks from BGP routing information to study the utilization of IPv4 address space. These CIDR blocks represent the routable portion of the IPv4 address space. The studies show a dense allocation of the classical Class-C addresses. They also show that the allocation of IP addresses in Class-A and Class-B address spaces tends to prefer the beginning of the address ranges. These studies, however,

do not quantify the probability of reachability for those address blocks.

## V. SUMMARY

Measurement projects and services that require discovering reachable addresses within an AP either direct their measurements to random hosts within the AP [11], [14], or use external sources, such as Web logs or Web crawler [5], [3], to identify some reachable hosts. Such external sources are limited in the number of reachable hosts they can provide.

In this paper, we developed and evaluated a set of heuristics to discover a reachable host within an AP. The IP selection heuristics work by selectively examining IP addresses within each AP. A number of IP addresses are selected for probing based on the required accuracy for AP discovery. These IPs are chosen based on empirical observations of their probability of being assigned to a host. One can use Figure 8 to determine the appropriate number of probes required to achieve a given discovery accuracy. The first 11 probes used in our “Conditional probing” method discover more than 90% of the active APs. We have also presented heuristics to prioritize passive APs to probe. We showed that using our methodologies, more than 62% of active APs can be discovered with only a single probe, and more than 92% of them can be discovered using less than 12 probes.

## REFERENCES

- [1] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy,” RFC 1519, Internet Engineering Task Force, Sept. 1993.
- [2] Y. Rekhter and T. Li, “A border gateway protocol 4 (bgp-4),” RFC 1771, Internet Engineering Task Force, Mar. 1995.
- [3] V. N. Padmanabhan and L. Subramanian, “An Investigation of Geographic Mapping Techniques for Internet Hosts,” *Proc. of ACM SIGCOMM*, 2001.
- [4] P. Francis et al., “IDMaps: A Global Internet Host Distance Estimation Service,” *ACM/IEEE Transactions on Networking*, Oct. 2001.
- [5] B. Krishnamurthy and J. Wang, “On Network-Aware Clustering of Web Clients,” *Proc. of ACM SIGCOMM 2000*, pp. 97–110, Aug. 2000.
- [6] Cooperative Association for Internet Data Analysis (CAIDA), “IPv4 Address Space Utilization,” url: <http://www.caida.org/outreach/resources/learn/ipv4space/>, 1998.
- [7] Hans-Werner Braun, “BGP-system Usage of 32-bit Internet Address Space,” url: <http://moat.nlanr.net/IPaddrocc/>, Nov. 1997.
- [8] C. Huitema, “The H Ratio for Address Assignment Efficiency,” RFC 1715, Internet Engineering Task Force, Nov. 1994.
- [9] G. Huston, “Commentary on Inter-Domain Routing in the Internet,” RFC 3221, Internet Engineering Task Force, Dec. 2001.
- [10] R. Siamwalla, Sharma R., and S. Keshav, “Discovering Internet Topology,” url: <http://london.ensim.net/keshav/papers.html>, July, 1998.
- [11] R. Govindan and H. Tangmunarunkit, “Heuristics for Internet Map Discovery,” *Proc. of IEEE INFOCOM 2000*, 2000.
- [12] “Game of Battleship,” <http://www.centralconnector.com/GAMES/battleship.html>.
- [13] NLANR, “National laboratory for applied network research routing data,” <http://moat.nlanr.net/Routing/rawdata/>.
- [14] H. Burch and B. Cheswick, “Mapping the Internet,” *IEEE Computer*, vol. 32, no. 4, pp. 97–98, April 1999.
- [15] B. Cheswick, H. Burch, and S. Branigan, “Mapping and Visualizing the Internet,” *Usenix 2000 general conference, San Diego*, June 2000.
- [16] “Skitter,” <http://www.caida.org/tools/measurement/skitter/>.
- [17] B. Krishnamurthy and J. Wang, “Topology Modeling via Cluster Graphs,” *ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.