IP Geolocation in Metropolitan Area Networks

Randolph Baden
University of Maryland, College Park

ABSTRACT

Existing techniques can geolocate an IP address to a metropolitan area. Through simulation, we evaluate the performance of these existing techniques within a metropolitan area network. We identify differences between metropolitan area networks and the wide area network. We describe and evaluate new techniques which are designed specifically for use on metropolitan area networks. We present Hop-Based Geolocation, a geolocation technique that is effective under certain network topologies when *ping* latencies are dominated by processing delay rather than propagation delay. We show that geolocation techniques based only on *ping* latencies and *traceroute* paths are not yet precise enough for their intended applications.

1. INTRODUCTION

Given an IP address, how accurately can we identify that geographical location of the machine with that IP address? Existing techniques [4, 5, 7, 8] can geolocate an IP address to a metropolitan area, but are unable to narrow down the location further. Ideally, we would like to be able to identify the street address for a given IP address, but this may not be possible with the information that we can gather from the network. Specifically, we attempt to solve the following problem: can geolocation techniques designed for the wide area network (WAN) be applied on metropolitan area networks (MANs) to improve the precision of those algorithms?

High-precision IP geolocation is useful for several applications. Geolocation assistance could be provided to 911 services for calls placed using VoIP. Laptop owners could install software which publishes the IP address and location of their laptop periodically to assist in recovery after a theft. Malicious Internet activity which can only be traced back to an IP address could be investigated more easily by the proper authorities. As with WAN geolocation [7], advertising could be targeted even more specifically to the appropriate recipients. With each of these applications, the primary barrier to their deployment is the precision and accuracy that we can obtain through IP geolocation.

Unfortunately, MAN geolocation is difficult for many reasons. Fundamental differences between MANs and the WAN suggest that techniques which are applicable on the WAN may not be appropriate for a MAN. Due to the limited information to which we have access without the explicit assistance of ISPs, there are theoretical limits to our precision and accuracy. Deploying a MAN geolocation system is expensive, so it is difficult to test MAN geolocation in a real network.

The remainder of this paper proceeds as follows. Section 2 outlines the differences between MANs and the WAN and how they are relevant to geolocation. Section 3 describes in detail existing and new geolocation techniques. Section 4 evaluates these techniques in simulation, and Section 5 presents questions and insight that arise in the evaluation. Section 6 identifies the limitations of our evaluation, and Section 7 concludes.

2. MANs

Existing geolocation techniques locate IP addresses in a Wide Area Network (WAN). These techniques typically only provide resolution at the city level. For applications such as regional advertising or default language selection, this level of resolution is sufficient. However, there are applications which would benefit from geolocation being as precise as possible.

Due to the fundamental differences between a MAN and the WAN, applying existing geolocation techniques on a MAN is not guaranteed to significantly improve the accuracy or precision. These differences can be broadly characterized as differences in *scale*, and differences in *topology*.

In terms of scale, MANs involve far fewer machines and cover a much smaller geographical region. MANs typically are between 5 km and 50 km in diameter. While queuing and processing latencies still affect RTTs in a MAN, the propagation delay is negligible[1]. It may be more difficult to deploy the necessary machines at geographically diverse vantage points, while options such as PlanetLab [1] are already available on the WAN.

MANs consist primarily of residential broadband networks. Residential broadband networks usually operate at Layer 2 of the OSI model, so *traceroute* information can be lacking, incomplete, or misleading. In particular, for customers on most residential broadband networks, there is a Layer 2 channel from the customer's modem to one of the gateway routers, as shown in Figure 1. The channel is used for all incoming and outgoing IP traffic, regardless of whether that traffic is to or from another machine in the same network. If this is the case, a MAN will be a small cluster of star

[1] 1 ms of RTT corresponds to approximately 200 km along a straight copper wire

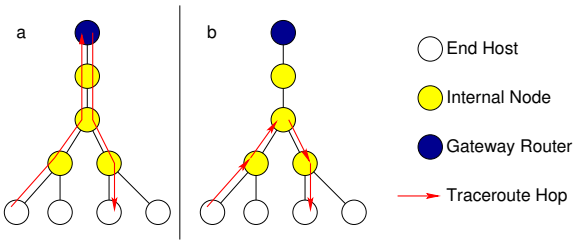

Figure 1: Traceroute information in a network with a) Layer 2 Permanent Virtual Circuits (PVCs) or b) Layer 3 IP routing.

topologies (Figure 2a), and *traceroute* will provide far less information than it does on a WAN.

As video-on-demand (VoD) services become more popular, residential broadband networks may adopt a Layer 3 architecture in order to support IP multicast. Such MANs would appear through *traceroute* to be more like small clusters of tree topologies (Figure 2b). Unlike the WAN, the scale of these edge networks allows them to be more tightly organized into tree topologies rather than meshes with many redundant links. Even though *traceroute* provides significant information in this case as it does in the WAN, the differences in the structure of the MAN and WAN topologies may affect the performance of the geolocation techniques presented in Section 3.

3. TECHNIQUES

Existing geolocation techniques are broadly divided [7] into two categories: semantic approaches and probing approaches. Semantic techniques infer information about an IP address based on DNS queries or a *Whois* lookup. DNS records may provide geographical information in the DNS name for an IP address, or they may provide specific geographical information for an IP address, such as its latitude and longitude. Geographical information in the DNS name is limited to the city level, so it is not precise enough to further refine a search within a MAN. Also, we would like a solution that works for general IP addresses (specifically, for client machines), so we do not expect latitude and longitude information to be available via DNS. Similarly, *Whois* will provide information at the city level, and for most IPs it will only provide the name of the ISP that controls the IP's address block.

Instead we focus on trying to apply probing geolocation techniques on a MAN. Many types of information are available through probing, including but not limited to *ping* latencies, *traceroute* paths, queuing delay correlation, and available bandwidth. The techniques we explore in this paper only take advantage of *ping* latencies and *traceroute* paths. Some techniques only require the RTT estimates from *ping*, while others also incorporate network topology information from *traceroute*. As described in Section 2, techniques which require *traceroute* are not always available within a MAN, but residential broadband network trends may change in the future.

The node with the IP address we try to geolocate is called the *target*. For these techniques, we require a set of *probe* nodes, which are nodes with known locations which can send *ping* and *traceroute* messages to each other and to the target. Some techniques may also benefit from having a set of *landmarks*, which are nodes with known locations but which do not send *ping* or *traceroute* messages. While landmarks do not provide as much information as probe nodes, we also do not need to install any software on them, which greatly eases deployment. Recursively applying a geolocation algorithm in a MAN network involves first using a global set of probes and landmarks to identify the MAN in which the target lies, and then using a local set of probes and landmarks within that MAN to reduce the search space further.

For each algorithm, the input to the algorithm is the *ping* and *traceroute* data from the probe nodes to each other, to the landmarks, and to the target. The outputs of the algorithm are a point and a region in geographical space. Certain algorithms guarantee that the target will lie in the reported regions; others only expect the target to lie in the reported region. Using expectation rather than guarantees allows a trade-off between the region size (*precision*) and the probability that the target is actually contained within the region (*accuracy*).

We describe three existing probing techniques: Constraint-Based Geolocation (CBG) [4], Octant [8], and Topology-Based Geolocation (TBG) [5]. We also describe our extensions of the existing algorithms, and present a new technique called Hop-Based Geolocation (HBG).

3.1 Constraint-Based Geolocation

The goal in the existing CBG algorithm is to narrow down the search space as far as possible while still guaranteeing that the target is contained inside. To do so, each probe node independently determines the maximum possible distance d to the target, and reports a circle (projected on the surface of the Earth) centered around its location having radius d. Since the target must be contained within all of the circles, the target must also be contained within the intersection of the circles. The final reported region in the algorithm is that intersection, and the final reported point is the centroid of that region. An example is presented in Figure 3.

For this algorithm to work, there must be a way to estimate the maximum distance d. CBG uses the latencies observed by the probe nodes to construct a latency-to-distance mapping. A simple example which in fact will guarantee that the target is contained within the region is to use the *baseline* mapping. The *baseline* mapping is given by $d(l) = \frac{2}{3}c * l$, where $\frac{2}{3}c$ is the speed of light on a copper wire. For a given latency l, this function computes the maximum distance (bounded by physical possibility) to the target assuming that l consists entirely of propagation delay and that the target lies on a "straight" path along the surface of the earth.

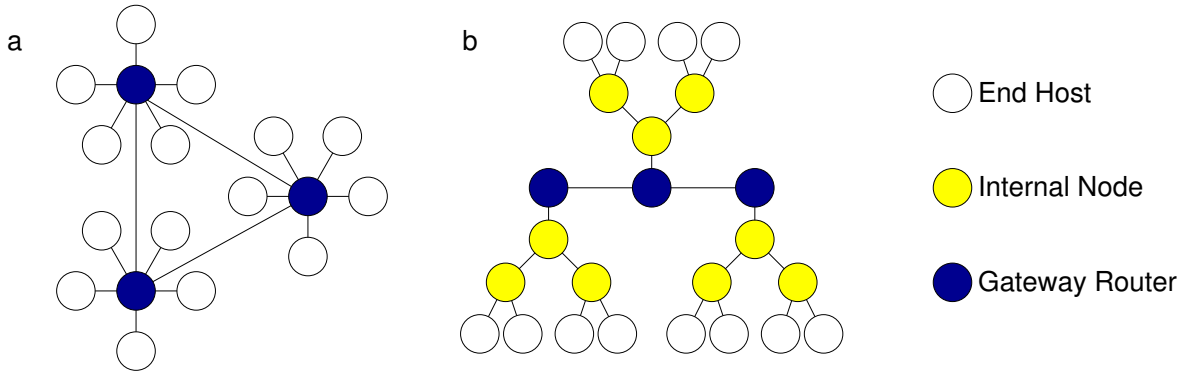

Figure 2: MAN topologies visible to *traceroute* when internal networks are a) Layer 2, or b) Layer 3.

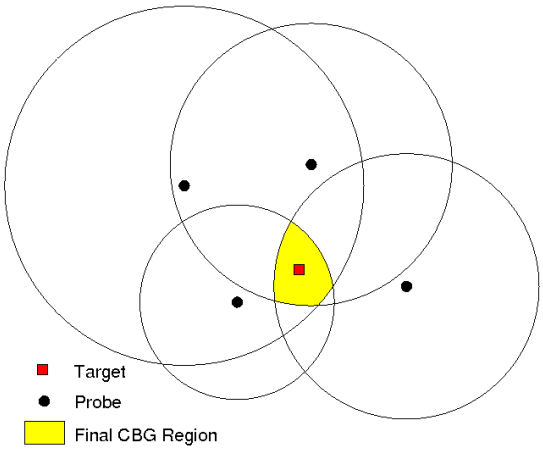

Figure 3: CBG with four probes.

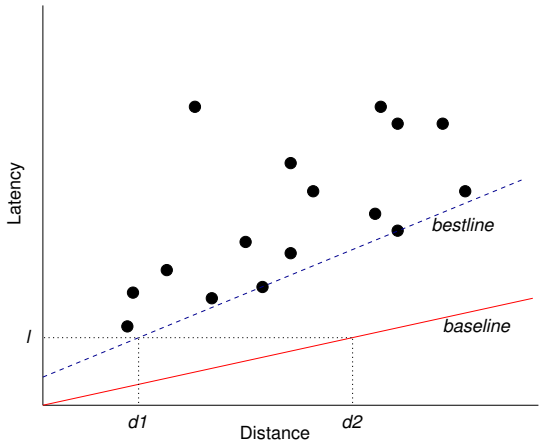

Figure 4: Finding estimated distance for latency l with latency-to-distance mappings. The *bestline* mapping restricts estimated distances based on observed data points ($d1$), but the *baseline* mapping is constant and independent of the data ($d2$).

In practice, physical paths are not straight and the latency is not only determined by the propagation delay, so CBG does not use the *baseline* mapping. Each probe machine instead constructs a *bestline* mapping by creating points for each other probe machine. The points consist of the distance and observed latency to those machines. The *bestline* is the line that lies entirely below the points but minimizes the sum of the distances from the line to the points. Figure 4 shows the *bestline* and the *baseline* for a single probe machine.

Using the *bestline* mapping for CBG results in a smaller region, but no longer guarantees that the target is contained in the region (for instance, if the network path to the target is substantially straighter than the network paths to the probe nodes). When the region is restricted to the point where it is possible for the target to lie outside the region, we say that the region is *overconstrained*. Overconstraining is not necessarily bad; in fact it is necessary to overconstrain in order to improve the precision of the algorithms beyond *baseline* CBG.

Though we cannot use landmark nodes to create additional constraints in CBG, we can use them to generate more data points for the *bestline* mappings. Doing so improves the accuracy of the mappings by sampling more of the network.

CBG as described at this point is presented entirely in [4]. One modification which we impose on CBG (and all of the other algorithms as well) is to handle faults gracefully. We sometimes overconstrain to the point that the intersection of constraints is empty. Rather than report that the algorithm has failed entirely, we remove constraints (beginning with the smallest one by area) until the intersection is non-empty.

We next describe three extensions to CBG, which apply to geolocation on both MANs and the WAN.

3.1.1 CBG_2

CBG_2 extends the basic CBG algorithm by creating additional "virtual" probe machines. After the initial probing between the probe machines and to the target, we select a router on the *traceroute* path to the target from one of the probe machines. The probe machines then probe that router. After this, we have all of the latency information necessary for that router to be a probe machine; the only thing we lack is the geographical location of that router. Rather than fix a point as the location of the router and construct a circle

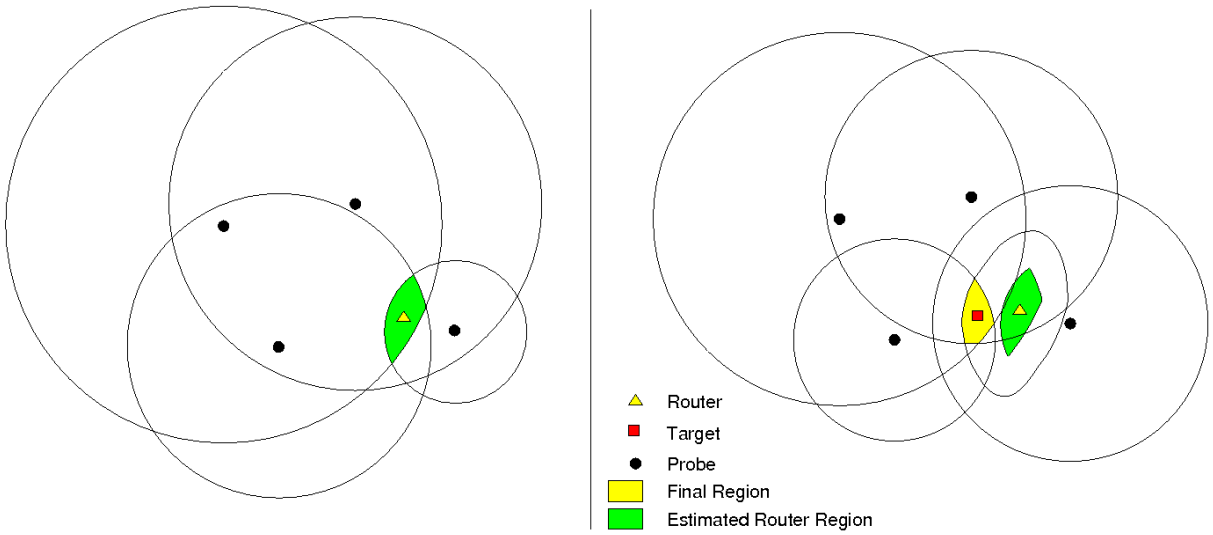

Figure 5: CBG_2: using CBG to find an intermediate router (left), and using CBG with the router as a virtual probe (right)

around it, we use *baseline* CBG to estimate the region in which that router lies, and extend the boundary of that region by the maximum distance to the target from that router according to the *bestline* mapping. Figure 5 shows this process when incorporating a single virtual probe machine.

Intuitively, we expect that if we repeat this process enough, we will reduce the effect of zigzagging paths to the target. However, it is possible that the additional constraints from the virtual probes will not add any new information if *baseline* CBG does not predict their locations precisely enough. It is also possible that the constraints for the virtual probes which use the *bestline* mapping can lead to overconstraining.

3.1.2 CBG_3

CBG_3 attempts to improve the accuracy of the *bestline* mapping. Since our goal with the mapping is to try to accurately estimate how distance corresponds to latency in the part of the network that contains the target, we restrict the *bestline* mapping to only use data points from the probes which share the most hops in the *traceroute* path with the target. There is a trade-off between the number of data points in the mapping and the relevance of the data points to the target; too few data points can lead to overconstraining, but irrelevant data points may make our constraints looser than they should be. Although we only use a subset of the probes for our data in the *bestline* mappings, every probe still participates in the algorithm and generates its own mapping.

3.1.3 CBG_4

One might expect that the latency to a destination is not based only on distance, but also on direction. CBG_4 uses a 3-dimensional mapping between (latitude, longitude) and latency, rather than the 2-dimensional *bestline* mapping. We divide the space into a grid, and associate with each cell of the grid the k nearest probes. In our experiments we use $k = 5$; we discuss our choice of parameters in Section 6. We then compute the *bestline* mapping for that cell similarly to the basic CBG algorithm. The final region for a given probe is the union of the cells in which the target is expected to lie; we take the intersection of the regions from all of the probes as the final reported region in the algorithm. This algorithm requires a large number of probes or landmarks spread throughout the space to build an accurate mapping.

3.2 Octant

Octant [8] is another existing technique that imposes even more constraints than CBG. Rather than just place maximum distance constraints on the target's location, each probe also places minimum distance constraints. Octant uses the upper and lower convex hulls of the points in the latency-to-distance mappings instead of the *bestline* mapping. Octant overconstrains the system more than the *bestline* mapping does, and the reported region is an intersection of annular regions rather than circles. Like CBG, only the latency-to-distance mappings benefit from landmarks in Octant.

3.2.1 APBG

We build upon Octant in Annular, Probability-Based Geolocation (APBG). Rather than generate a single annular region, we generate some fixed number n^2 of annular regions for each probe node. These regions are disjoint, and the union of all of the regions makes up the *baseline* CBG circle for that probe. Each region is assigned a weight based on the probability that a machine at latency l away (where l is observed through *traceroutes* to the target) lies in that region. The probability is estimated by the observed distribution of probe nodes, landmarks, and previous target locations, so having a large number of landmarks is useful for APBG.

[2]We use $n = 3$ in our experiments

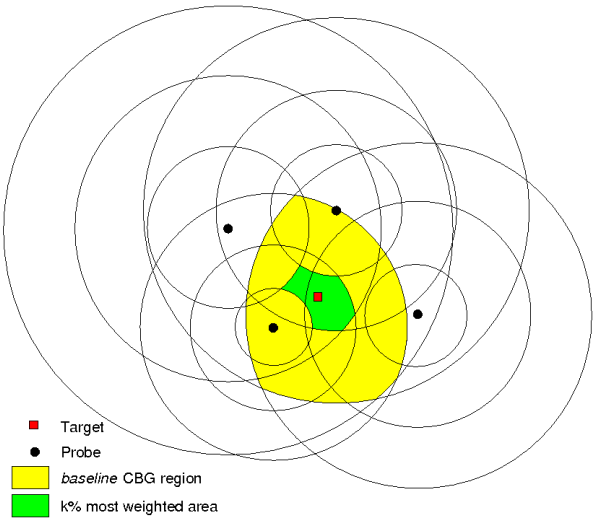

Figure 6: APBG with 4 probes and 3 annular regions.

If p is the number of probe nodes, then when we combine the information from the probes, we have potentially p^n regions in which the target may lie. In practice we have far fewer since we discard the regions which do not intersect at least one annular region from every probe. We combine the weights and take at least the top $k\%$[3] of the total area as the final region, and the centroid of that region as the final point. It is not clear how best to combine the weights if we use the probabilities from the distributions; for simplicity we multiply the weights to combine them. This final region is a subset of the *baseline* CBG region by construction. Figure 6 presents a visualization of the algorithm.

3.3 Topology-Based Geolocation

Rather than creating a system of constraints to find a region where the target lies, the Topology-Based Geolocation (TBG) algorithm attempts to solve for the location of all of the routers and the target simultaneously. We use Vivaldi [2] as a solver for this system of equations.

We assume that there are straight line paths between the routers in the *traceroute* paths and (perhaps erroneously on a MAN) that the main source of latency is propagation delay. Initially, we assign to each router in the *traceroute* paths a random geographical location within the MAN; the algorithm will iteratively move the router to a more appropriate location.

During each iteration of the Vivaldi algorithm, we assign an error value to each link between two IP addresses seen through *traceroute*; our goal is to minimize the sum of the absolute value of these error values. This error value is the difference between the observed latency (obtained through *traceroute*) and the estimated latency. The estimated latency is the latency between the end points of the link based on their estimated locations and a global *bestline* mapping which incorporates all of the latency and distance pairs from all of the probe nodes. The two end points of a link are pulled towards each other or pushed away from each other according to the error value. By reducing the distance by which the end points move on successive iterations, we ensure that the system reaches a stable solution. We are not guaranteed that the solution is a global minimum, so we run the algorithm multiple times with different initial positions and use the solution with the smallest error sum as our final solution.

Unlike standard Vivaldi, our system is slightly simpler. Since the coordinates of the probe machines are known, those coordinates never change and the routers move relative to the probes. We do not need to account for pathological behaviors in Vivaldi such as a rotating coordinate system, since the probe nodes anchor the coordinates to the desired location. Figure 7 shows several iterations of this algorithm.

The final location of the target is given by its coordinates in the final solution. TBG does not actually report a region in which the target lies; to compare TBG to the other algorithms we perform *bestline* CBG from the last hop router(s) for the target using the estimated locations of those last hop routers. If the target has only one last hop router, we report a circle around the last hop router.

TBG can benefit significantly from landmarks. Each landmark is also considered to be an anchor (like the probes). The closer that the target is to an anchor in the network, the better we expect our estimate to be.

TBG as described at this point is presented in [5] and [2]. We present an extension of TBG next.

3.3.1 TBG$_2$

The TBG$_2$ algorithm improves upon TBG by incorporating two pieces of information from *baseline* CBG. First, it uses the reported location of the target from *baseline* CBG as the initial estimate of the target's location, rather than a random location in the space. Unless the target is near an anchor in the network, it may be far from its actual location in the space in a stable solution. We use the CBG estimate to ensure that the target is near its actual location to begin with.

We also use the fact that the region reported by *baseline* CBG must contain the target. As before, we run Vivaldi multiple times (with different initial positions for the routers, not the target) and report the most successful solution. A solution is immediately discarded if the target does not lie within the region reported by *baseline* CBG.

3.4 Hop-Based Geolocation

Based on the observation that distance has very little effect on latency on a MAN scale, we propose a new technique called Hop-Based Geolocation (HBG). If propagation delay is negligible, and we minimize queuing delay with repeated measurements, then all that remains is serialization and processing delay. Both forms of delay are based on the number of hops along the Layer 2 and Layer 3 path, of which we can only see the Layer 3 part with *traceroute*.

In the HBG algorithm each probe machine sends *traceroute* messages to each other probe machine. The first goal in

[3]We use $k = 10$ in our experiments

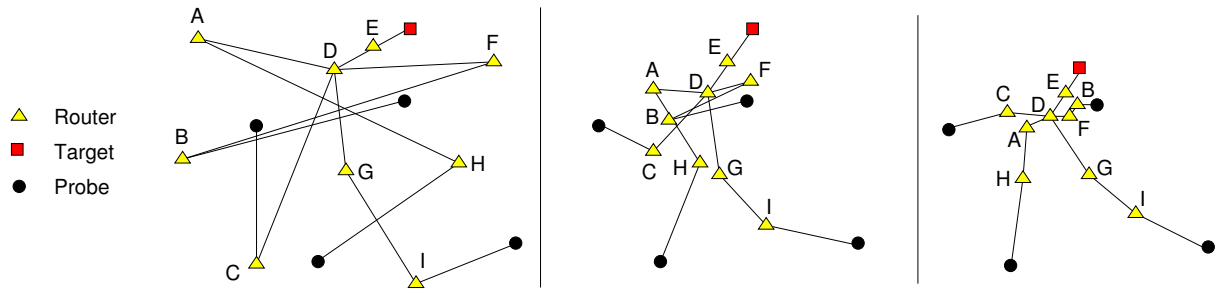

Figure 7: Three iterations of TBG. The final result depends on the randomly chosen initial positions.

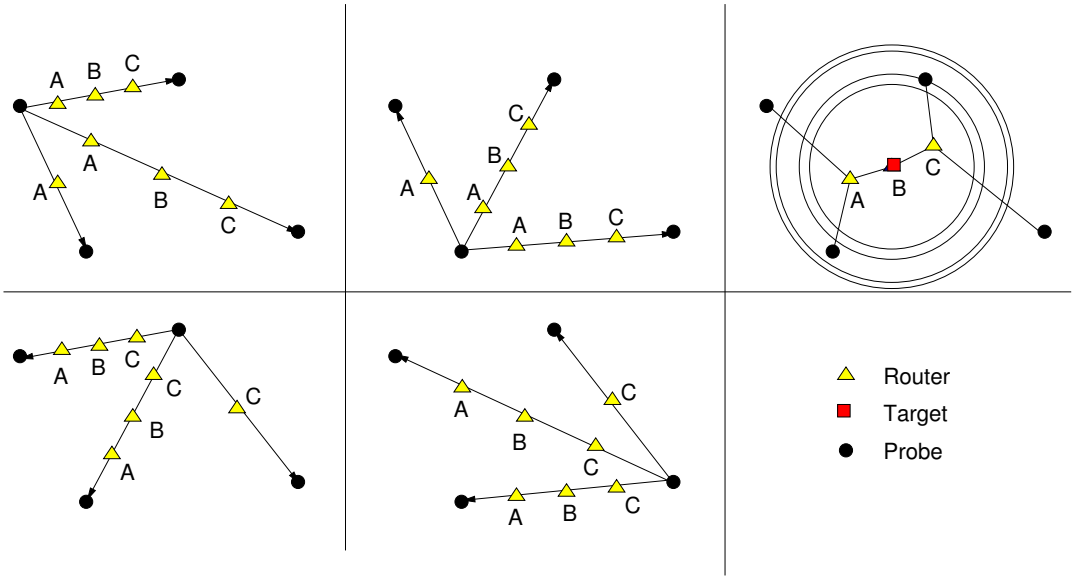

Figure 8: HBG with 4 nodes. Each probe estimates B's location identically, but differ slightly on the size of the bounding circle, so the intersection provides the smallest circle. In all cases, the location of B (the last known router location) is the reported location of the target, even though the target is not colocated with B.

HBG is to estimate the locations of all of the routers in these *traceroute* paths. Since we want to avoid using the latency information from *traceroute*, for each *traceroute* message we estimate that the routers on the path lie equally spaced on a straight line between the sender and receiver. Over all of the *traceroutes* we may see a single router multiple times, so we average all of the estimates of a router's location together to get a final estimate of the router's location. Figure 8 shows an example of this process.

In addition to estimating the router locations, we also construct a hop-to-distance mapping for each probe. These mappings are identical to the *bestline* mapping in CBG, only we use the number of hops as an indicator of distance instead of the latency. We use these hop-to-distance mappings to estimate the distance from an intermediate router to the target. However, the points in the mappings represent distances between end hosts, so it is a potential limitation of this approach that these mappings may not directly apply to routers in the middle of the network.

After estimating router locations and building hop-to-distance mappings, each probe sends *traceroute* messages to the target. Each probe finds the hop nearest to the target for which it has an estimated location, and reports a circle around that hop. The circle has a radius given by the hop-to-distance mapping, based on the number of hops between the router and the target. Finally, we intersect the circles to get the reported region, and take the centroid as the reported point.

Like TBG, HBG benefits from having landmarks to act as additional endpoints of *traceroute* paths. Also, like CBG, *traceroutes* to the landmarks improve the accuracy of the hop-to-distance mappings.

We present a summary of all of the geolocation techniques in Figure 9.

4. EVALUATION

Existing techniques have been evaluated on the WAN on testbeds such as PlanetLab[1]. To our knowledge there are no publicly available testbeds specifically on individual MANs, so we resort to simulation. We evaluate the techniques on

Technique	Requires *traceroute*	Over-constrains	Landmark Benefit
baseline CBG	No	No	None
bestline CBG	No	Yes	Small
CBG_2	Yes	Yes	Small
CBG_3	Yes	Yes	Small
CBG_4	No	Yes	Large
Octant	No	Yes	Small
APBG	No	Yes	Large
TBG	Yes	Yes	Large
TBG_2	Yes	Yes	Large
HBG	Yes	Yes	Large

Figure 9: Summary of properties of probe-based geolocation techniques.

two classes of network topologies.

We use the GT-ITM Transit-Stub model [9] to generate the first network topology, which we call MiniWAN. The GT-ITM model generates WAN topologies, so we do not expect that the topologies will be appropriate for a MAN. However, we place the nodes in geographical space on a MAN scale, so we test the techniques in a setting where the propagation delay is negligible. The algorithm for assigning node positions is as follows. Beginning with the backbone routers (in the transit ASes), we place nodes randomly in the space if we have not placed any of the routers to which they are linked. If we have placed at least one of the routers to which they are linked, we choose one of those routers arbitrarily and place the new router in a random direction between 0.2 km and 12 km away. We use 5 transit ASes with 4 stub ASes per transit node, and have a total of 1620 nodes in the system. We assume that there is a constant 1 ms of processing delay between connected nodes, a propagation delay according to the straight line path between the end points, and that there is no queuing delay (since repeated measurements could reduce its effect). Our choices for parameters are arbitrary (or based on intuition) in this simulator; this constructs a MAN that could potentially exist, not a representative one.

The second type of network topology we generate is a single DSL network, which we call DSLNet. The network consists of three types of nodes: customers, DSLAMs, and core nodes. Due to the hierarchical nature of the DSL network, we are able to generate topologies with 50,000 customer nodes, 250 DSLAMs, and 50 core nodes. First, we place each core node and DSLAM randomly throughout a 20 km by 20 km square. Then, we assign to each customer a DSLAM and place that customer within a 5 km radius of it. Each customer creates links to its assigned DSLAM, each DSLAM creates links to its nearest core node, and the core nodes create links to each other in a mesh where nearby core nodes are more likely to be connected than faraway ones. We assume latencies between connected nodes as in the first topology with the following exception: links between customers and DSLAMs have a 3 ms processing delay since the last hop is a major source of latency for DSL networks [3]. We believe that this topology generator better models a typical MAN, but it still does not capture the diversity of MAN topologies.

4.1 Effect of Topology

To gauge how important topology is to the performance of the geolocation techniques presented in Section 3, we compare the results obtained through both of our topology generators. For both topology generators, we test 24 topologies randomly generated as described above. On each of these topologies, we geolocate 30 targets using every technique described in Section 3. For this experiment, we do not use any landmark nodes. We use a subset of only the 20 closest probes (by latency) of the total set of 500 probe nodes for each search attempt for scalability (which we discuss in Section 6).

We present the results in Figure 10. We make the following observations:

- Error distances and region sizes are much larger on the MiniWAN topologies. Since we do not constrain the geographical region for the MAN to lie in a 20 km by 20 km square, we expect this to be the case. Despite this, the relative performance of the algorithms remains the same in most cases.
- Success rates are higher in general on MiniWAN topologies than on DSLNet topologies.
- CBG algorithms are in general far more accurate (have higher success rates) on MiniWAN topologies than on DSLNet topologies.
- TBG algorithms report very large regions in the MiniWAN topologies, but very small regions in the DSLNet topologies.
- HBG is more precise and accurate than the other algorithms which constrain the region significantly. This is more pronounced in the DSLNet topology, but it is also true in the MiniWAN topology.

4.2 Landmarks and Probes

We now consider our DSL network topology generator alone to see how the algorithms perform when we increase the number of landmarks and probes. We perform the same experiment with 10 landmarks (while keeping 20 probes), and with 30 probes (while keeping 0 landmarks).

We present the results in Figure 11. We make the following observations:

- Using either more probes or more landmarks significantly improves the accuracy and precision of CBG_2. We examine this result more closely in Section 4.3.
- Using more probes or landmarks does not significantly alter the performance of the TBG algorithms, Octant, APBG, or HBG.

	MiniWAN (20 probes, 0 landmarks)			DSLNet (20 probes, 0 landmarks)		
	Avg. Error	Avg. Region	Success Rate	Avg. Error	Avg. Region	Success Rate
baseline CBG	**4.90 km**	35514.07 km^2	**100%**	3.36 km	431027.47 km^2	**100%**
bestline CBG	6.95 km	264.33 km^2	55%	5.18 km	8.05 km^2	18%
CBG_2	6.86 km	92.79 km^2	46%	3.83 km	46.46 km^2	28%
CBG_3	8.16 km	243.56 km^2	51%	5.06 km	7.37 km^2	16%
CBG_4	20.16 km	2151.20 km^2	**100%**	5.65 km	249.61 km^2	**100%**
Octant	6.91 km	**28.78 km^2**	22%	4.20 km	1.37 km^2	12%
APBG	6.84 km	348.08 km^2	31%	3.48 km	38.57 km^2	31%
TBG	48.02 km	1262.23 km^2	8%	3.83 km	**0.04 km^2**	4%
TBG_2	45.75 km	1222.25 km^2	9%	2.92 km	**0.04 km^2**	6%
HBG	6.85 km	59.79 km^2	54%	**2.81 km**	3.28 km^2	41%

Figure 10: Precision and accuracy of geolocation techniques on MiniWAN and DSLNet. Error is the distance from the reported point to the actual location of the target, Region is the area of the reported region, and Success Rate indicates whether the target was contained in the reported region. Very large region sizes may be inaccurate due to the projection of the Earth's surface into a 2-dimensional space.

	DSLNet (30 probes, 0 landmarks)			DSLNet (20 probes, 10 landmarks)		
	Avg. Error	Avg. Region	Success Rate	Avg. Error	Avg. Region	Success Rate
baseline CBG	3.28 km	568370.37 km^2	**100%**	3.39 km	692281.80 km^2	**100%**
bestline CBG	5.49 km	5.01 km^2	17%	4.11 km	5.56 km^2	31%
CBG_2	4.16 km	7.88 km^2	40%	3.88 km	8.37 km^2	43%
CBG_3	5.75 km	5.51 km^2	14%	4.68 km	7.14 km^2	24%
CBG_4	12.06 km	60.91 km^2	26%	9.79 km	133.57 km^2	55%
Octant	4.34 km	1.19 km^2	14%	3.99 km	1.18 km^2	13%
APBG	3.55 km	46.58 km^2	33%	3.70 km	35.86 km^2	34%
TBG	3.69 km	**0.05 km^2**	3%	4.05 km	**0.05 km^2**	3%
TBG_2	2.90 km	**0.05 km^2**	5%	3.04 km	**0.05 km^2**	4%
HBG	**2.80 km**	3.00 km^2	41%	**2.83 km**	3.37 km^2	44%

Figure 11: Precision and accuracy of geolocation techniques on DSLNet with more probes and landmarks.

4.3 Individual Algorithms

While average error distances and region sizes are useful metrics for gauging the precision of a geolocation algorithm, they can be inflated by a small number of very inaccurate outliers. We present the performance of CBG_2, TBG_2, and HBG in more detail in order to better understand their performance.

Figure 12 shows the CDFs of CBG_2 error distances and region sizes. Here, we see that region sizes are usually much smaller when using only 20 probe machines (rather than 30 probes or 20 probes and 10 landmarks). This indicates that the average region size from Figure 10 is inflated when compared with the corresponding results in Figure 11, due to a small number of outliers with very large region sizes. We expect that the accuracy rate is lower when we report smaller regions; Figures 10 and 11 erroneously indicate that the opposite is true but Figure 12 shows that our intuition is correct.

On the other hand, Figure 13 shows that TBG_2 actually performs as Figures 10 and 11 describe. The MiniWAN topologies do have very large error distances and region sizes, while the DSLNet topologies have small error distances and very small region sizes. Unlike CBG_2, TBG_2 can have inaccurate results even with large region sizes since the region is centered around a point very far away from the target. We discuss why TBG and TBG_2 behave so differently on MiniWAN and DSLNet in Section 5.

Since HBG appears to perform well, we present its results in more detail in Figure 14. We see that HBG consistently reports regions between 2 and 4 km^2 in area in the DSLNet topologies. Since we only know that the target must lie within 5 km of its last hop router, these regions are still far smaller than the "optimal" region that we could predict if we knew the location of the last hop router. We have erred in favor of precision in the trade-off between precision and accuracy.

5. DISCUSSION

MAN geolocation is a difficult problem. In this paper, we present several algorithms which intuitively seem like they would solve the problem well. Due to our incomplete understanding of MAN topologies and the limited theoretical precision in geolocation on a MAN, some intuitive algorithms actually perform very poorly, while non-intuitive algorithms perform well. Although this is more pronounced on MANs,

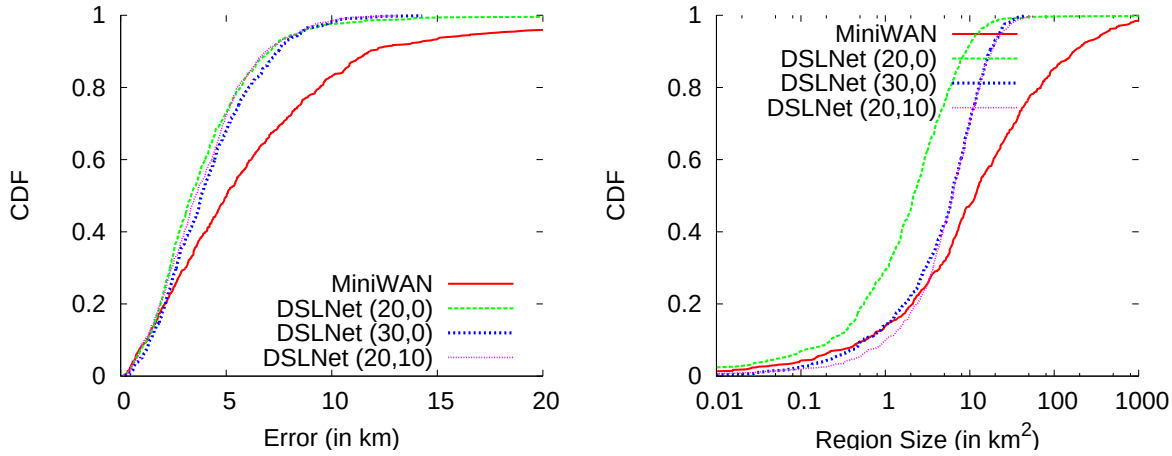

Figure 12: CDF of CBG_2 error distances (left) and region sizes (right).

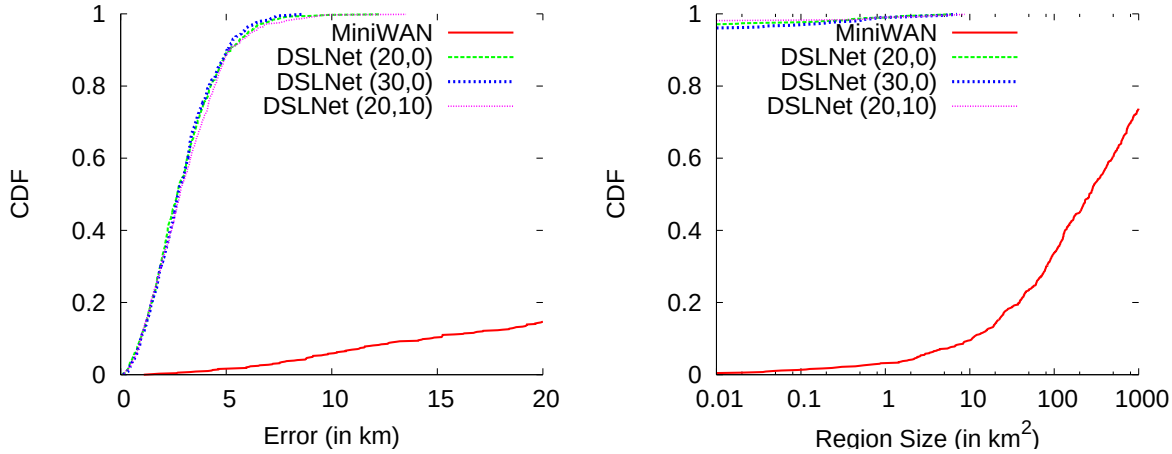

Figure 13: CDF of TBG_2 error distances (left) and region sizes (right).

Katz-Bassett *et al.* [5] acknowledge similar behavior on the WAN. We do not claim to completely understand why all of the algorithms perform the way they do, but we discuss some observations we made in our experiments.

It is necessary to overconstrain in MAN geolocation to produce meaningful results. If we base our constraints on the *baseline* when propagation delay is negligible, the reported region is extremely large. One possible alternative which we have not considered is to also infer tighter minimum bounds on the processing delay at each router, in order to attempt to isolate the propagation delay as much as possible. It is unclear whether this is even possible, or whether it is sufficient to extract the propagation delay to the point where it is no longer negligible.

CBG region sizes do not necessarily decrease as we add more constraints. We adopt the policy of pruning constraints when our intersection of constraints is empty. In WAN geolocation, empty intersections are far less common since the *bestline* mapping is more accurate when propagation delay is meaningful. In MAN geolocation, something must be done to account for such faults; though we choose the simple solution of removing the tightest constraints first, we have not explored alternatives. By adding additional constraints which cause an empty intersection, we may counter-intuitively end up with a larger region than we had before we add the constraint. Figure 15 shows an example of how this may happen.

CBG_4 is very imprecise. While we believe that this is due to an insufficient number of data points in the latency-to-distance mappings for each grid cell, we have not been able to simulate enough landmarks to show that this is in fact the case. Even if we have enough data points, we still expect the *bestline* mappings to be inaccurate in CBG_4 as they are in CBG due to negligible propagation delay.

Octant reports small final regions, but is very inaccurate. The convex hull of the data points in the latency-to-distance mappings is even more restrictive than the *bestline* mapping.

The TBG algorithms are far too imprecise in the MiniWAN experiments, and far too precise in the DSLNet exper-

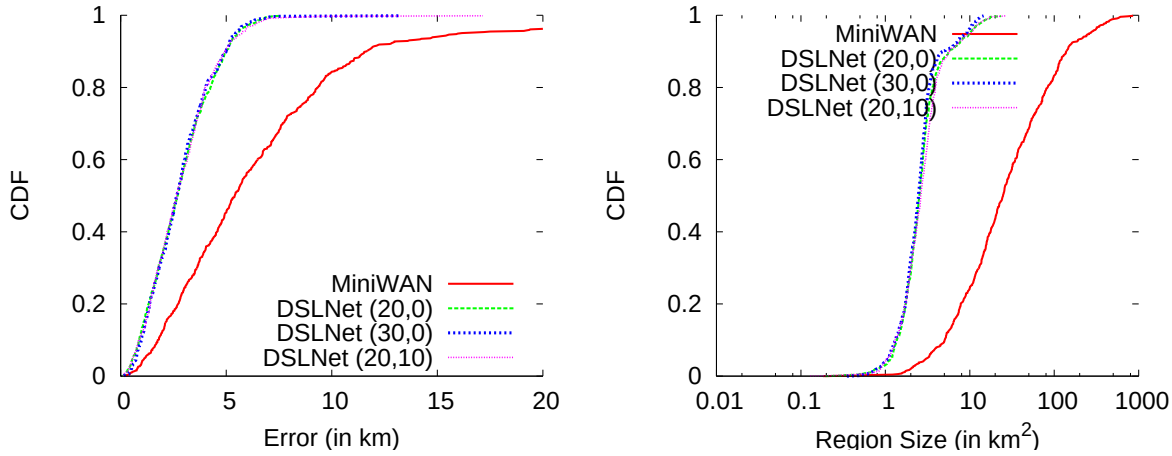

Figure 14: CDF of HBG error distances (left) and region sizes (right).

iments. In the MiniWAN experiments, the Vivaldi system is unstable and has high error values, so the router coordinates expand outwards away from the anchors, resulting in high error distances. In the DSLNet, the system is stable and the error distances are similar to other geolocation algorithms. The region size cannot be explained through the stability of the Vivaldi system, since it depends only on the latency from the last hop router to the target and the global *bestline* mapping. We believe that the vast difference in region sizes between the MiniWAN and DSLNet topologies are due to our processing delay model. Specifically, in the DSLNet topology, the delay between the last hop and the target is approximately 3 ms, while it is approximately 1 ms in the MiniWAN topology. While the global *bestline* mapping may have data points for 1 ms of latency which may be as far as 12 km away in the MiniWAN topology, the smallest latency for which we have a data point in the DSLNet topology is 6 ms (and it is guaranteed to be within 10 km). We observe flatter *bestline* mappings with high y-intercepts in the MiniWAN topologies, and sloped *bestline* mappings with low y-intercepts in the DSLNet topologies.

HBG in particular performs well despite the simplicity of the algorithm. We have no intuitive reason why straight lines between end points of traceroute paths would estimate the location of the routers well, but in practice it does. We believe that this may be due to the tree-like structure of the DSLNet topologies, but we need to investigate more to understand this phenomenon.

Looking at the performance of the algorithms in general in Figures 10 and 11, we see that the level of precision and accuracy which we can obtain through MAN geolocation is probably not sufficient for high-precision applications, which we expect to require more than 80% accuracy and less than 2 km of error. This is a fundamental shortcoming of geolocation techniques which just rely on *ping* latencies and *traceroute* paths. In the future we intend to consider other measurement options specific to the MAN scale (such as queuing delay correlation and bandwidth attenuation) to see if we can improve precision further.

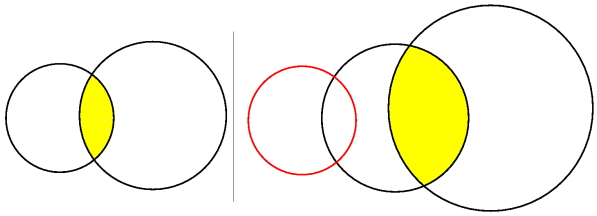

Figure 15: Adding an extra constraint can lead to a larger intersection region given our policy of pruning small constraints when we have an empty intersection.

6. LIMITATIONS

The limitations of this work are divided into two areas: limitations of the simulator and general geolocation limitations. Despite these limitations, we feel that this is the first step towards solving the problem of MAN geolocation. The next step is to establish MAN test-beds to evaluate geolocation techniques in real networks in order to remove the effect of simulator limitations. We mention the limitations here with the intent of removing these potential sources of error in the future.

6.1 Simulator Limitations

6.1.1 MAN Topologies

We attempt to represent a realistic MAN topology in our simulator. In practice, not only will MAN topologies differ from what we have presented here, but they will also differ from each other since the term MAN covers a diverse category of networks. The simulator could be improved by incorporating real MAN topologies discovered through *traceroute* and by physically locating end hosts and routers to provide realistic geographical coordinates for as many nodes in the system as possible.

6.1.2 Scalability

The total number of *traceroute* measurements required by the geolocation algorithms is quadratic in the number of probe machines. The geolocation techniques also have their own running times and memory overhead. Altogether this limits the number of searches and number of probes which we can test on a single network topology. We believe that we have collected a sufficient amount of data to have a meaningful evaluation. In practice, since the system would be distributed, we could use many more probes and landmarks. This would benefit some algorithms more than others, but we are unable to determine the extent of that benefit with our current simulator.

6.1.3 Queuing Delay

We ignore the problem of queuing delay in simulation. The impact of queuing delay should be negligible if we take sufficiently many measurements, but in practice it may not be possible to take enough measurements if we want to locate the target quickly.

6.1.4 Parameters

Although we try to choose appropriate parameters for our geolocation algorithms and the topology generators, we do not know the best set of parameters. Over the course of our evaluation, we tried to select values which gave the best performance, but we have yet to rigorously test each parameter individually. It is possible that we could do better with slightly different parameters, but we do not expect the improvement to be substantial.

6.2 Geolocation Limitations

6.2.1 IP Visibility

Since many MANs primarily use Layer 2 rather than IP routing, *traceroute* might provide far less information in practice than we use in our simulator. This is a general limitation of all of the geolocation techniques that use *traceroute*.

Even if the MAN network uses IP routers, all of the intra-network traffic may still go through the gateway router. This creates a single common intersection in all of the *traceroute* paths, which limits the effectiveness of certain techniques such as TBG or HBG. Also, if a router's interfaces have different IP addresses (IP aliases), we may not be able to identify when *traceroute* paths intersect, which is also detrimental.

6.2.2 Aware Target

If the person running the target machine is aware that they are being geolocated, they can evade geolocation [6]. A simple method of doing so is to inflate latency measurements by delaying ping responses. In that case, we can only at best estimate the location of their last hop router(s). We may or may not be able to put bounds on the distance between the target and the last hop router(s) depending on the underlying technology used in the network. In DSL networks, for instance, customers must be within approximately 5 km of their DSLAM.

6.2.3 Deployment

For the most part, we have ignored the issue of the actual cost of deploying a system to perform geolocation on a metropolitan area. To be able to perform geolocation on a single MAN requires a large number of probe machines within that MAN. Providing a service that requires accurate geolocation in all of the major metropolitan areas in the United States, for instance, would be very expensive. For this reason alone, it is unlikely that widespread services requiring MAN geolocation will be feasible unless an existing infrastructure could supply the probe machines.

7. CONCLUSION

We have evaluated the precision and accuracy of existing and new geolocation techniques on MANs in simulation. Despite the limitations of the simulator, our experiments provide a starting point for future evaluation on real MANs. In particular, we describe HBG, a new geolocation technique which performs better than existing algorithms when *traceroute* measurements are available. We show that MAN geolocation (using only *ping* latencies and *traceroute* paths) is not precise or accurate enough to be used for high-precision applications.

8. ACKNOWLEDGMENTS

This work is sponsored by the Laboratory for Telecommunications Sciences. This project was made possible by the assistance of Satinder Pal Singh, Mark Shayman, Richard La, and Bobby Bhattacharjee.

9. REFERENCES

[1] B. Chun, *et al.* PlanetLab: An Overlay Testbed for Broad-Coverage Services. *ACM SIGCOMM Computer Communication Review*, 33(3):00–00, 2003.

[2] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A decentralized network coordinate system, 2004.

[3] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM Press, New York, NY, USA, 2007.

[4] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based geolocation of internet hosts. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 288–293. ACM, New York, NY, USA, 2004. ISBN 1-58113-821-0.

[5] E. Katz-Bassett, *et al.* Towards ip geolocation using delay and topology measurements. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 71–84. ACM, New York, NY, USA, 2006. ISBN 1-59593-561-4.

[6] J. Muir and P. van Oorschot. Internet geolocation and evasion. Tech. rep., Carleton University, 2006.

[7] V. N. Padmanabhan and L. Subramanian. An investigation of geographic mapping techniques for internet hosts. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 173–185. ACM, New York, NY, USA, 2001. ISBN 1-58113-411-8.

[8] B. Wong, I. Stoyanov, and E. G. Sirer. Geolocalization on the internet through constraint satisfaction. In *WORLDS'06: Proceedings of the 3rd conference on USENIX Workshop on Real, Large Distributed Systems*, pp. 1–1. USENIX Association, Berkeley, CA, USA, 2006.

[9] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *IEEE Infocom*, vol. 2, pp. 594–602. IEEE, San Francisco, CA, 1996.