Dynamic Landmark Triangles: A Simple and Efficient Mechanism for Inter-Host Latency Estimation [*]

Zhihua Wen
Electrical Engineering & Computer Science
Case Western Reserve University
zhihua.wen@case.edu

Michael Rabinovich
Electrical Engineering & Computer Science
Case Western Reserve University
misha@eecs.case.edu

ABSTRACT

This paper describes a simple and efficient approach to estimate the network latency between arbitrary Internet hosts. We use three landmark hosts forming a triangle in two-dimensional space to estimate the distance between arbitrary hosts with simple trigonometrical calculations. To improve the accuracy of estimation, we dynamically choose the "best" triangle for a given pair of hosts using a heuristic algorithm. Experiments using several data sets of measured host-pair latencies, as well as a live Internet study, demonstrate the accuracy and efficiency of our approach.

1. INTRODUCTION

The modern Internet is experiencing a rapid growth in large-scale distributed applications utilizing overlay and peer-to-peer networks. The performance and scaling properties of these applications crucially depend on exercising communication paths according to network distances between hosts. Furthermore, the high scale of these systems often stipulates hierarchical design, which typically relies on host clustering based on network proximity. For example, content delivery networks, such as Akamai and ICDS by AT&T, use distance estimation to direct clients to the closest edge servers for content download. In peer-to-peer file sharing applications like BitTorrent or Gnutella, each peer can benefit by selecting the closest peers to itself. Network clustering is fundamental to hierarchical P2P systems, such as Gnutella 2, where super peers represent clusters of neighboring hosts, and to large-scale Internet characterizations and monitoring [2, 12, 13]. Unfortunately, the scale of these systems often makes direct on-demand distance measurements between hosts impractical.

A number of approaches have been proposed to handle this problem by predicting the distance rather than measuring it directly. Different approaches are tailored for different applications. In particular, IDMaps [7] aims at mapping inter-host distances on the global Internet scale. King [9] has a unique ability to estimate distance between uncooperative hosts, even if they do not respond to probes. Coordinate-based techniques [26, 25, 28, 19, 31, 3, 4] map hosts to points in a metric space. These techniques are especially well-suited in applications that reuse host coordinates for multiple tasks since these tasks can be accomplished without further costs. Meridian [35] addresses the problems related to server selection, approaching them not through prediction but through actual measurements between the target host and exponentially narrowing set of server candidates.

In this paper, we propose an approach that targets applications that do not require full network positioning of hosts but only inter-host distances. This includes a wide range of applications such as overlay topology formation, server selection, and node clustering mentioned earlier. Our approach is extremely simple but, as we show through extensive experiments, is accurate and efficient at this task. Unlike full network positioning techniques, we employ only three landmarks, forming a triangle in a two-dimensional Euclidean space, to estimate the distance between a pair of end hosts using simple trigonometrical calculations. However, we use a relatively large number of *potential* landmarks and carefully select only three of them for any given prediction. Having a large number of potential landmarks allows us to select the landmarks for each prediction that produce high accuracy estimates. Using a small number of landmarks for actual probing and computations keep the overheads low. Thus, our approach represents a tradeoff of achieving high accuracy and low overhead at the expense of having to deploy a relatively large number of well-dispersed potential landmarks.

Our estimation involves two rounds of probes. In the first round, we use a general, big triangle to obtain a rough distance estimates between each end host and the landmark candidates. We then use these rough estimates as the basis for selecting another landmark triangle, which is likely to obtain a high accuracy distance estimation for the current end hosts. This second triangle produces the final distance estimation in the

[*] This work is supported by the National Science Foundation under Grant No. 0721890. The approach of the present paper was initially outlined in a position paper [33].

second round of probing.

We used a three-pronged approach to evaluate our method and compare it with existing techniques. First, we evaluate it from the general prediction accuracy perspective. Second, we study its performance in the context of some specific overlay applications - host clustering and server selection. Third, we performed a a live Internet study comparing the quality of server selection in the Akamai content delivery network [1] using our approach and GNP, which was previously proposed for a similar task [30].

Our results indicate that our simple technique is accurate and efficient at the tasks it targets. For instance, in the live Akamai study, our approach showed a better selection quality than GNP with 1/7-th less probing traffic. Thus, while different approaches are best-suited for different applications, we believe our method will be a valuable addition to the arsenal of overlay application designers.

2. RELATED WORK

A number of approaches to inter-host distance estimation have been proposed, with different approaches better suited for different applications. The King utility [9] estimates the distance between two arbitrary hosts by the distance between their authoritative DNS servers. It is more accurate for well-connected hosts than residential hosts, which typically have high latency to their DNS servers [34]. The IDMaps [7] platform utilized a large number of tracer hosts at strategic locations in the Internet and estimated the distance between two end hosts as the sum of the distance between each host and its nearest tracer plus the distance between the two tracers. IDMaps's goal was to map rough inter-host distances on the global Internet scale within a factor of two of the real distances. The approach by Hotz [10] exploits the observation that if triangle inequalities held on the Internet, then the distance between two end-hosts would be bounded between the difference of the distances from the end-hosts to a common landmark and the sum of these distances. We discuss the IDMaps and Hotz's approaches further in Section 5.3.

Starting with the pioneering Global Network Positioning (GNP) approach [26], a number of current techniques are based on embedding hosts into a multidimensional metric space. Their key advantage is that once a host's coordinates are computed, they can be reused for distance estimation to any other mapped hosts with no additional overhead. GNP assigns coordinates to a node based on the measured distance from this node to a fixed set of well known hosts called landmarks. GNP is considerably more accurate than IDMaps and King. However, owing to its use of simplex downhill to solve a multidimensional nonlinear minimization problem for error minimization, GNP incurs high computational overhead unless an application can effectively amortize it. Subsequent approaches [3, 25, 28] have dealt with a number of important issues such as security and landmark load but used the same underlying estimation principle.

Some approaches, including Virtual Landmarks [31] and the Internet Coordinate System [19], utilize *Lipschitz embedding* of hosts into a Euclidean space and replace simplex downhill with an efficient linear approximation based on the principal component analysis. In particular, the Virtual Landmarks study [31] reported similar accuracy to GNP, although on our data sets, this approach achieved its speedup at the expense of accuracy loss.

Vivaldi [4] is a coordinates-based approach that requires no fixed landmarks. Instead, each node constantly adjusts its own coordinates by communicating with other peers and following a mass-spring relaxation model whose minimum energy state determines the node coordinates. Vivaldi incurs no probing cost in its targeted peer-to-peer applications since it derives its measurements from regular P2P communication. We include Vivaldi in our performance study. Big Bang Simulation (BBS) [29] is a somewhat similar approach which models the network as a set of particles under the effect of the potential force field to determine node positions.

Recently, the suitability of Euclidean embedding for distance prediction has been questioned. In particular, Lua et al. [20] showed that these schemes appear to have poor accuracy under a new *relative rank loss* metric, which the authors introduced as more meaningful to a number of applications. Lee et al. [16] attributed this issue to a large incidence of violations of triangle inequality on the Internet. We include relative rank loss as one of the general evaluation criteria of our approach.

Unlike coordinate-based approaches, Meridian [35] does not employ distance estimates but targets the server selection applications through direct measurements. Each node in Meridian organizes other nodes into rings of different radius, with a fixed number k of nodes in each ring. To find the closest node to a given target host, Meridian starts from a random node, and explores a small set of nodes that are in the current node's rings and at about the same distance from the current node as the target. The query is then forwarded to the closest node to the target, and the search terminates when the current node is the closest. According to the Meridian study [35], its server selection is an order of magnitude more accurate than that of coordinate schemes such GNP or Vivaldi. However, the query process takes $O(logN)$ sequential forwarding steps and $O(klogN)$ probes, where N is the total number of servers. According to [35], a system with 2000 servers and $k = 16$ required just over 10K probing traffic per query on average, or over a hundred probes.

Two recent studies report on the experiences with utilizing network distance estimations in real applications. Ledlie et al [14] analyze the implementation of the Vivaldi algorithm in Azureus, a popular Bit-Torrent client, and Szymaniak et al [30] report on utilizing GNP in the Google content delivery network. We return to the latter paper in Section 7.2.

Most distance estimation systems collect their measurements by repeated probes between each pair of hosts and use the median or minimum value[26, 4] as the true network distance. Ledlie et al. [15] introduce techniques to reduce the necessary probing traffic between pairs of hosts. Our approach can also benefit from their techniques.

3. LANDMARK TRIANGLES

Our method estimates the distance between a pair of hosts using their distance to three landmark hosts, and improves the accuracy of this estimate by selecting the most appropriate landmarks for the given hosts. As all distance prediction techniques, our method uses a round-trip latency as the distance metric, which could be obtained using direct probes, such as ping or tcpping, or through passive measurements. Depending on the application, the distances could be measured from a host to landmark or in the other direction. We consider these and other architectural details (e.g., who and when computes the distance estimates) when we discuss specific applications but simply assume the availability of RTT measurements in the general algorithm.

3.1 Basic Algorithm

Our basic idea is illustrated in Figures 1 and 2. Consider for a moment that hosts are mapped to a two-dimensional Euclidean space, and we can measure true deterministic distances between the hosts. Assume we know the distance from two fixed points L1 and L2 to a point A. This limits point A to only two possible locations, A' and A'', situated symmetrically around line (L1,L2) as shown in Figure 1. Hence, for two points A and B, the known distance from each of them to points L1 and L2 limits the distance between A and B to only two possible values, depending on whether A and B are situated across or on the same side of line (L1,L2). We can use simple trigonometric calculations, described in the Appendix, to compute these distances.

Considering Figure 2 now, we can differentiate between these two possibilities with the help of a *witness* point L3 located off the (L1,L2) line. Indeed, by comparing the known distance from L3 to either point with the distances to the point's two possible locations, we can tell if either point and L3 are across or on the same side of line (L1,L2). If the two points are both on the same side with L3, or both on the opposite side from L3, the two points are on the same side with each other; otherwise (if one point is on the same side with L3 and the other on the opposite side) the two points are across line (L1,L2).

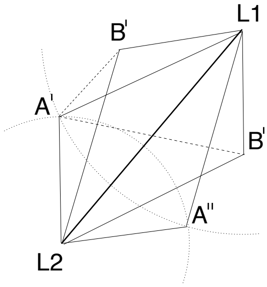


Figure 1: Possible locations of points A and B given the distance to landmarks L1 and L2.

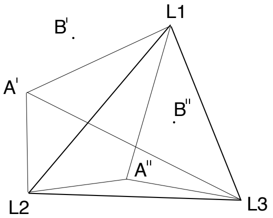


Figure 2: Using a witness landmark (L3) to disambiguate possible locations of points A and B.

Thus, we can unambiguously compute the distance between points A and B from the distances from A and B to the fixed points L1, L2, and L3. We call points L1 and L2 calculation landmarks and point L3 witness landmark.

Unfortunately, Internet hosts do not map precisely to a Euclidean space. We deal with one major consequence of the imperfect mapping, which is triangle inequality violations, in the next subsection. Another consequence is that the measured distance from the witness L3 to, say, point A is unlikely to match either of the two possible values, given by the length of segments (L3,A') and (L3,A"). Fortunately, we are only using L3 to distinguish between two discrete possibilities and not to compute the distance estimate; thus, we simply choose between the two possible positions of point A depending on which of the two possible values is closer to the measured distance from L3 to A. In order to improve the accuracy, we repeat our calculation by using landmarks L1 and L2, L2 and L3, and L1 and L3, as calculation landmarks, with the remaining landmark used as the witness. We then take the average of the three estimates as the final value.

3.2 Triangle Inequality Violations

Several previous studies observed that hosts on the Internet can violate the triangle inequality [37, 36, 21].

These violations may lead to a situation when the measured distances between hosts and landmarks allow no possible mapping of hosts into the Euclidean plane.

Recall that a given landmark triangle generates three combinations of calculation and witness landmarks, and we normally use the average of the three corresponding distance results as the final distance estimate. When some pairs of the calculation landmarks yield no consistent host mappings, we simply consider only the remaining pairs that do produce such mappings. In the extreme case, when all three pairs of calculation landmarks exhibit triangle inequality violation with one of the end hosts, we use the minimum of the three measured indirect paths between the end hosts through each of the three landmarks as the estimated distance. The degree of the triangle inequality violations and its effect on distance estimation accuracy is analyzed in Section 5.

It might appear that a more natural way to handle the above case is to pick another landmark triangle, in the hope it would not violate the triangle inequality. However, this would cost another round of network latency measurements between the new landmarks and each end-host, which would be detrimental in a real time distance estimation system that our approach targets.

4. DYNAMIC TRIANGLE SELECTION

It might appear natural to improve the triangles-based approach by increasing the number of landmarks and using random selection of possible triangles to obtain a number of estimates, which could then be processed with standard statistical methods. However, we found that this actually had a detrimental effect on accuracy. The reason is that the prediction error in different triangles is not random: some triangles are “good”, i.e., they produce highly accurate predictions and some are “bad”. Adding more random triangles does not necessarily improve the quality of the mix. It turns out that one can characterize triangles in terms of the likely accuracy of their predictions. The next section examines these heuristics and uses them to develop an algorithm for dynamic triangle selection.

Our general approach to improve accuracy of distance estimation is to have a large set of landmarks that could *potentially* be used in a landmark triangle and to select a specific triangle for a particular pair of hosts. Our goal then is to develop a heuristic to dynamically select a landmark that is likely to produce a high-accuracy estimate.

4.1 Intuition

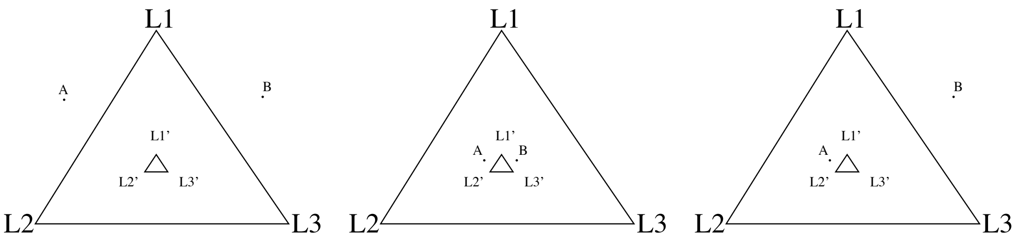
We develop our heuristic for dynamic triangle selection based on the following intuition. Consider Figures 3(a), 3(b), and 3(c), where $L1$, $L2$ and $L3$ form a big triangle, $L1'$, $L2'$ and $L3'$ form a small triangle, and we want to estimate the distance between point A and B. On Figure 3(a), the points A and B are far apart and both are far away from the small triangle. Then, from the position of points A and B, the three landmarks $L1'$, $L2'$ and $L3'$ approximate one single point. Thus, the accuracy of this small triangle is very poor. For the big triangle of $L1$,$L2$ and $L3$, the distances between the landmarks themselves and between points A and B are comparable, so using big triangle in such a case can generate a relatively good estimate.

Turning to Figure 3(b), both points A and B are close to the small triangle and hence close to each other. From the view point of landmarks $L1$,$L2$ and $L3$, point A and point B look like a single point, so the accuracy using the big triangle is relatively poor in this case. But for the small triangle of landmarks $L1'$, $L2'$ and $L3'$, the inter-landmark distances are comparable to the distance between points A and B. Thus, using the small triangle in this case can generate a better estimate.

Finally, on Figure 3(c), the points A and B are far apart but one of the points (point A) is close to the small triangle. The big triangle in this case can generate relatively good estimates similar to Figure 3(a). However, using the small triangle can also generate good results in this situation. Indeed, although its landmarks $L1'$, $L2'$ and $L3'$ all look like one point to B, since point A is very close to this small triangle, the triangle estimation in this case essentially approximates the distance between points A and B by the distance between point B and the landmarks $L1'$, $L2'$ and $L3'$. Because these landmarks and point A are all close to each other, the accuracy of this estimation is high. By selecting a small enough triangle that is close enough to one of the end-hosts, we can obtain a better estimate than the estimate produced by the big triangle.

Interestingly, in the last case, the small triangle might be so close to point A that one could accurately approximate the distance between A and B directly by the distance between B and the triangle’s vertex closest to A, without using the other two vertexes at all. This would save probing from the other two vertexes as well as trigonometric calculations. We leave this optimization (including a tricky part, which is a heuristic to decide when using a single landmark is acceptable) to future work.

Overall, a small triangle is likely to generate poor accuracy when it is far removed from both end-hosts and high accuracy when it is close to at least one of the end-hosts, regardless of the distance between the end hosts. For hosts that are far apart and without a nearby small triangle, a big triangle is more likely to produce decent accuracy. Thus, our heuristic for dynamic triangle selection starts with a general big triangle (i.e., the same for all host-pairs), and then progressively replaces it



(a) Both points A and B are far away from the small triangle (b) Both points A and B are very close to the small triangle (c) Point A is close to the small triangle and B is far away from the small triangle

Figure 3: The effect of the landmark triangle position and size on the accuracy of inter-host distance estimation.

with triangles that are *both* smaller *and* closer to one of the end-hosts.

4.2 Triangle Selection Algorithm

Following the above observations, we have designed a heuristic algorithm for dynamic triangle landmark selection shown in pseudocode in Figure 4. For given hosts A and B, the algorithm tries to find a small triangle close to either of them and use it for distance estimation. If it cannot find such a triangle, it resorts to the general big triangle. The algorithm takes as input the distance between both end-hosts and every potential landmark. We discuss how to estimate these distances without extra measurements in Section 4.4.

The algorithm addresses two main issues. First, for a given host pair, it needs to find the smallest triangle, which is also the closest to either host. We combine these two selection criteria by maintaining a single dynamic "quality threshold". Once any small close triangle is identified, we set the quality threshold to the maximum of the triangle size (measured as the maximum edge) and its distance to the closest host (measured as the distance from the host to the farthest vertex). We then replace this triangle with another triangle only if the latter is both smaller and closer than the current threshold. Note that the new triangle might actually be either bigger or more distant than the triangle it replaces as long as it reduces the threshold that defines the overall selection quality[1].

Second, the algorithm must deal with a vast search space. For example, 100 candidate landmarks can form 161700 triangles for each host pair. Thus, the algorithm uses several optimizations to cut the number of considered triangles dramatically. For example, if the distances from both hosts A and B to a landmark vertex i are already greater than the current threshold then,

```
// $N$ is the number of landmarks
// $D_{i,j}$ is the distance between landmarks $i$ and $j$
// $D_{A,i}$ is the distance between host $A$ and landmark $i$
// $T_{i,j,k}$ is the triangle formed by landmarks $i$ $j$, and $k$
// $m_T$ is the largest edge in triangle $T$
// $d_{A,T}$ is the max distance from host $A$ to any vertex in
// triangle $T$
$Selected\_Triangle = General\_Triangle$;
$Threshold = Threshold\_InitialValue$;
// Loop through all triangles but stop as soon as possible
$for\ (i\ from\ 1\ to\ N-2)\{$
    // If the distance from both hosts to the current landmark
    // already exceeds the threshold no need to proceed
    // with the loop
    $if\ (D_{A,i} > Threshold\ and\ D_{B,i} > Threshold)\ continue$;
    $for\ (j\ from\ (i+1)\ to\ N-1)\{$
        // If one edge of the current triangle already exceeds the
        // threshold no need to proceed
        $if\ (D_{i,j} > Threshold)\ continue$;
        $if\ (D_{A,j} > Threshold\ and\ D_{B,j} > Threshold)\ continue$;
        $for\ (k\ from\ (j+1)\ to\ N)\{$
            $if\ (D_{A,k} > Threshold\ and\ D_{B,k} > Threshold)$
                $continue$;
            $if\ (D_{i,k} > Threshold\ or\ D_{j,k} > Threshold)\ continue$;
            // A new small triangle is found; Reset the threshold
            $m_{T_{i,j,k}} = \max(D_{i,j}, D_{i,k}, D_{j,k})$;
            $m_{T_{i,j,k}} = MAX(D_{i,j}, D_{i,k}, D_{j,k})$;
            $d_{A,T_{i,j,k}} = MAX(D_{A,i}, D_{A,j}, D_{A,k})$;
            $d_{B,T_{i,j,k}} = MAX(D_{B,i}, D_{B,j}, D_{B,k})$;
            $Threshold = MIN(MAX(m_{T_{i,j,k}}, d_{A,T_{i,j,k}}),$
                            $MAX(m_{T_{i,j,k}}, d_{B,T_{i,j,k}}))$;
            $Selected\_Triangle = T_{i,j,k}$;
        }
    }
}
$RETURN\ Estimate(A, B, Selected\_Triangle)$;
```

Figure 4: Triangle selection algorithm.

[1] We currently do not consider triangle shape as part of the selection criteria. Intuitively, an equilateral triangle would be better than one with the three vertexes approaching a straight line. We will attempt to further improve our algorithm by accounting for triangle shapes in the future.

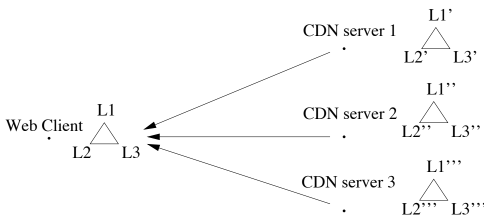


Figure 5: One-sided triangle selection in a CDN.

since we define the distance between a point and a triangle as the distance from the point to the triangle's farthest vertex, we need not examine *any* triangle with vertex i any more (line 4 in the algorithm). Similarly, if the distance between two landmark vertexes i and j is already greater than the current threshold, then we need not to consider any triangle with edge (i, j) since we use the maximum edge as the triangle size (line 8). Finally (and most importantly), once we find a triangle satisfying our requirement, i.e., with max edge and max distance to at least one point below the current threshold, we will use this triangle to set the threshold to a smaller value (line 17), which progressively removes further triangles from consideration. Even if the initial threshold is set to a very large value, it decreases quickly, resulting in a small number of triangles that end up being considered. For example, in our DZ-Gnutella data set (see Section 5.1), the average number of examined triangles for each host-pair reduces from 161,700 to less than 1000.

The initial threshold value presents a tradeoff: the smaller the value the lower the computational cost of triangle selection but the less likelihood that a small triangle will be found. We use a heuristic that attempts to approximate the goal of finding a small triangle for 90% of the end-hosts. Since we do not know the end-hosts in advance, we use the landmarks themselves instead and identify the initial value such that for 90% of landmarks, there exists a small triangle formed by the other landmarks. We found that, compared with the infinite initial threshold value, this heuristic results in negligible increase in estimate errors for a wide range of landmark numbers, while cutting significantly the computational cost of triangle selection. For example, in the DZ-Gnutella data set with 100 landmarks, the median relative error increases from 0.053193 to 0.053495 but the computation time to estimate the distances between all host pairs drops from 27.517s to 13.059s.

4.3 One-Sided Triangle Selection

The above algorithm is geared towards a scenario where one needs to estimate the distance between two hosts. It could be used, for example, when the two hosts respond to measurement probes such as pings or TCP pings but are otherwise uncooperative. In many applications, however, one needs to estimate the distance of an incoming host to a large set of exiting hosts. In these cases, the above algorithm may select different triangles for each new/existing host pair requiring a large number of probes. For example, in Figure 5, the server selection subsystem in a content delivery network might need to probe the client from triangle $(L1', L2', L3')$ to estimate the client's distance to server 1, from triangle $(L1'', L2'', L3'')$ to estimate its distance to server 2, and so on. This can easily negate an advantage of using the estimations over direct measurements.

Consequently, for these applications, we only choose between the general triangle and small triangles that are close to the new host. In the CDN example of Figure 5, the closest smallest triangle to the client, $(L1, L2, L3)$, will be used to estimate the distance from the client to all the servers. This obviously can lower the estimation accuracy but allows one to obtain the distance between the new and all existing hosts with only six probes – three from the general triangle and three from the selected client-specific triangle – regardless of the number of existing hosts involved.

4.4 Obtaining Host-to-Triangle Distance

The triangle selection algorithm utilizes distances between each host and all the candidate landmarks. Obtaining these distances by direct measurements, however, can generate significant probing traffic. Instead, we limit real measurements at the time of triangle selection only to measure distances between the hosts and the vertexes of the general big triangle. The distances to other candidate landmarks are themselves estimations using the big triangle.

Thus, the distance estimation between two hosts involves two rounds of probes and is as follows. The system maintains, asynchronously with distance estimations, the pair-wise measured distances between all the candidate landmarks. At the estimation time, we first measure the distances between the host pair under study and the general triangle vertexes. Second, we select a landmark triangle for the given host pair utilizing these distances to estimate the distance between each host and potential triangles under consideration. Third, we measure the distances between the selected landmark triangle and either host and use these measurements to estimate the distance between the hosts.

5. ACCURACY EVALUATION

This section presents the evaluation of the dynamic triangles approach from the general accuracy and overhead perspectives. We compare dynamic triangles with representative existing distance estimation techniques: GNP, Vivaldi, and Virtual Landmarks. When we apply our technique to server selection applications (Section 7.2) we add Meridian, an influential approach targeting these applications but not general distance estimations, into our analysis. We also briefly consider two earlier approaches, IDMaps and the Hotz's approach (Section 5.3). As mentioned in Section 2, other landmark-based schemes, while improving various aspects of GNP, exhibit similar accuracy. Also, GNP and King were compared previously in [34].

5.1 Methodology

We utilized four data sets in our study, one collected by ourselves and three available externally.

The *DZ-Gnutella* data set (available from [6]) was collected in June 2007 using DipZoom measurement infrastructure [5], which provides programmatic access to a number of measuring points (MPs) around the world from a locally run Java application. We asked almost 400 MPs across the globe to ping each other and a list of around 1900 Gnutella peers. We repeated these measurements until we collected over 22 million ping results. Similar to a number of previous studies [26], we used the minimum measured ping latency as the real distance between hosts. After removing hosts with few successfully collected distances, we ended up with 320 MPs and 1202 Gnutella peers. Most of DipZoom MPs ran on PlanetLab nodes but there were also 21 non-PlanetLab and residential nodes. We used these non-PlanetLab MPs separately in one of our experiments to further validate our results. Moreover, in most of the inter-host distances we study, one of the two hosts is an external (non-DipZoom) host.

We obtained the next two data sets from [24]. The *PL* data set contains the pair-wise latency distance among 226 PlanetLab nodes. It uses a median of several ping RTTs as the distance metric. The *PL-Azureus* data set, previously used in [14], contains the pair-wise distance among 248 PlanetLab nodes as well as the distance between these nodes and 2654 Azureus clients. This dataset uses active measurements from application-level communication, presumably over TCP. Finally, the *Meridian King* data set, available from [23] and previously used in [35], contains the distance between 2500 DNS servers collected using the King method[9]. The metric here is an estimated round-trip of UDP messages.

We now describe how we use these data sets to estimate accuracy of various distance estimate techniques.

For dynamic triangles experiments, we set aside 100 hosts that have a complete collection of pair-wise distances in the data sets, as candidate landmarks. When we find more than 100 such hosts, we select the 100 most widely distributed ones by first using k-mean clustering algorithm (based on inter-host distances) to partition the candidates into 100 clusters and then choosing, in each cluster, the closest host to its cluster center. We select the general big triangle among these candidates by partitioning them into three k-mean clusters and selecting a centroid host in each cluster as a triangle vertex. The remaining 97 candidates are available to form dynamic triangles, although some candidates may not participate in triangle selection for a given host-pair if the data set misses the measured distance between these candidates and either host.

For GNP, we used the software downloaded from [8] utilizing the default configuration with 15 landmarks and 8 dimension, denoted as GNP(15,8). Although the study [26] used 7 dimension, we assume the software reflects the latest recommendation of the GNP authors. We further tried different GNP configurations on the PL data set, including larger numbers of landmarks and higher dimensions, and confirmed that GNP(15,8) produced the best accuracy [32]. In each data set, we obtained the 15 GNP landmarks from the same 100 hosts used as landmark candidates for dynamic triangles, by first partitioning these hosts into 15 clusters with k-mean clustering and then selecting from each cluster the host closest to its center. We denote GNP with n landmarks and m dimensions as GNP-(n, m) in this paper.

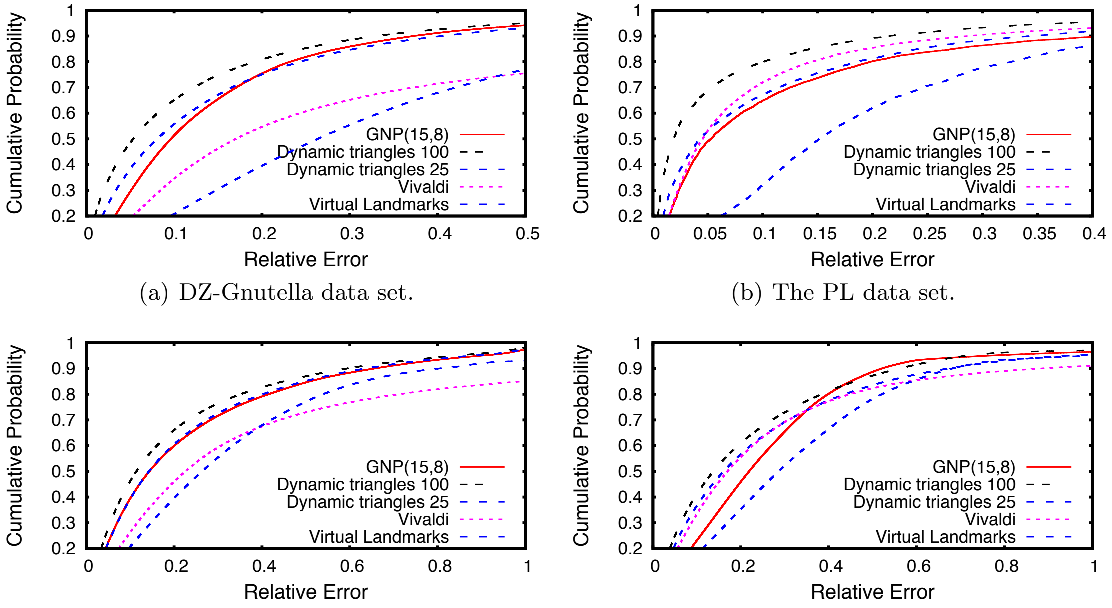
For Vivaldi, we downloaded the Vivaldi simulator from [24]. In all simulations, we set the neighbor count to 100 and, according to the suggestions in [4], utilized the height dimension. We ran each simulation until the coordinates of each node stabilize.

For Virtual Landmark (VL) experiments, in every data set, we used 20 landmarks and used PCA to reduce the number of dimensions from 20 to 8 – the same configuration as used in the original VL paper [31] to compare this approach with GNP. We again utilized our clustering approach to select the 20 landmarks.

We used the following metrics to evaluate the accuracy of estimates. Following the original studies of all the alternative techniques we consider, we used relative error to quantify the accuracy of our distance estimates defined as the difference between the predicted and measured distance divided by the smaller value among the two:

$$\frac{|predicteddistance - measureddistance|}{min(measureddistance, predicteddistance)}$$

Many applications, such as those involving server selection or overlay topology construction, depend not on the absolute accuracy of distance estimates but on the accurate ranking of hosts with respect to their distances to a given host. To assess the suitability of our approach to these applications, we use the *common closest peers* and *relative rank loss*, two rank preservation metrics introduced by Lua et al. [20]. Given a set of hosts S, the former measures the percent of the overlap between the k closest hosts from S to a given external host according to estimated and measured distances. The latter



(a) DZ-Gnutella data set.

(b) The PL data set.

(c) PL-Azureus data set.

(d) Meridian King data set.

Figure 6: Accuracy of distance estimates using dynamic landmark triangles.

measures the overall order preservation and is defined as a percentage of all host pairs in S that are ordered differently according to their estimated and measured distances to a given target host.

5.2 Relative Error of Distance Estimation

Figure 6 shows the cumulative distribution functions of the relative error of distance predictions using the estimated triangle selection (see Sec. 4.4) as well as existing approaches. Table 1 summarizes these results by listing median relative errors across all host-pairs. It also lists the average probing cost per end-host incurred in the experiments (not counting periodic landmark-to-landmark measurements, which add trivial cost[2]) and the total computational costs to obtain all estimates. We should note that real overhead can only be meaningfully discussed in the context of a particular application. For instance, the table shows high probing cost for Vivaldi while it incurs *no* such costs in its intended application where these measurements are observed passively. Similarly, the table shows high computation costs for GNP, which in some applications can be distributed to the end-hosts. At the same time, all probing cost numbers in the table assume that probes are reused for different estimates, which may not always be possible. We thus revisit overheads in Section 7 and only remark here that these general numbers indicate the flexibility of different approaches to being utilized in various contexts. Table 1 does not list computational costs for Vivaldi as they depend on the number of iterations one specifies and hence are not meaningful.

Focusing on accuracy, these results indicate that the quality of the dynamic triangles estimates is appreciably higher than existing approaches. Even with 25 candidate landmarks, when its general probing costs become similar to GNP and VL, their accuracy never fell below other techniques.

Another noteworthy observation is that the quality of Vivaldi estimates is much higher on PlanetLab data set than on the data sets involving Gnutella and AZureus clients. We think the reason is that Vivaldi works best on full matrix data set where each peer in the simulation can randomly select neighbors from all other peers. In the DZ-Gnutella and PL-AZureus data sets, a host can only select peers from the relatively few DipZoom MPs or PlanetLab nodes, respectively (because the data does not include distances between two Gnutella or AZureus clients). Thus, Vivaldi results on these two data sets may not represent its performance in its intended P2P applications where each client can measure the latency to any peer as long as it can connect to it.

We used the two-sample Kolmogorov-Smirnov test [11] to check statistical significance of the results in Figure 6. Given two empirical distributions, the K-S test uses the maximum difference between the two empirical CDFs for the same argument, $D^+ = \max_x |F_1(x) - F_2(x)|$ along with the size of each sample to produce the probability p of the hypothesis H_0 that both samples come

[2] E.g., with 100 landmarks, the total number of probes in one round of measurements is $(100 \quad 99)/2 = 4950$. Even refreshing these measurements every 10 minute means each landmark needs to send only one probe every 12 seconds.

Methods	Data Set	Median rel. error	Avg probes	Comp. time
Tri 100	DZ-Gnu	5.3%	55	13s
Tri 25		7.4%	13.5	7s
GNP		9.6%	15	9m
VL		23.5%	20	4 sec
Vivaldi		17 %	100	NA
Tri 100	PL	1.9 %	40	6s
Tri 25		4.2 %	14.4	5s
GNP		5.3%	15	8m
VL		15.1%	20	1s
Vivaldi		4.5 %	100	NA
Tri 100	PL-Az	11.4 %	59	106s
Tri 25		14.2 %	13.7	19s
GNP		14.3%	15	8m
VL		26.3%	20	10s
Vivaldi		22.6 %	100	NA
Tri 100	Meridian	13.8 %	51	228s
Tri 25		16.1 %	15.4	70s
GNP		21.9 %	15	504m
VL		28.6 %	20	42s
Vivaldi		17 %	100	NA

Table 1: Summary of estimation accuracy using different approaches on different data sets.

Methods	Data Set	D^+-value	p-value
Tri/GNP	DZ-Gnu	0.1941	$< 2.2e - 16$
Tri/VL		0.4306	$< 2.2e - 16$
Tri/Vivaldi		0.3105	$< 2.2e - 16$
Tri/GNP	PL	0.2571	$< 2.2e - 16$
Tri/VL		0.5367	$< 2.2e - 16$
Tri/Vivaldi		0.247	$< 2.2e - 16$
Tri/GNP	PL-Az	0.0685	$< 2.2e - 16$
Tri/VL		0.2759	$< 2.2e - 16$
Tri/Vivaldi		0.2101	$< 2.2e - 16$
Tri/GNP	Meridian	0.1297	$< 2.2e - 16$
Tri/VL		0.2614	$< 2.2e - 16$
Tri/Vivaldi		0.0786	$< 2.2e - 16$

Table 2: Statistical significance of the accuracy differences.

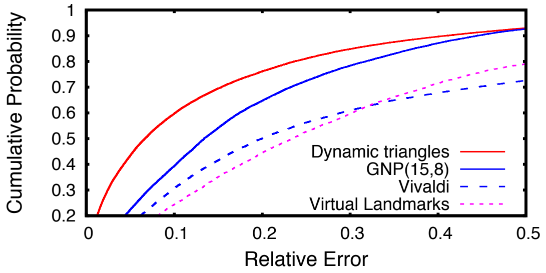


Figure 7: Estimate accuracy for non-PlanetLab MPs.

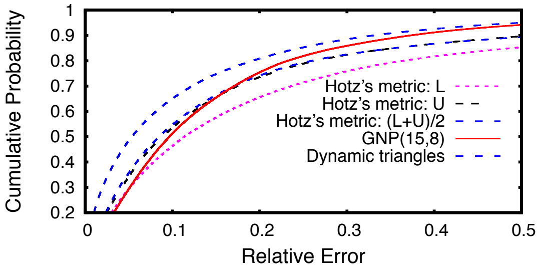


Figure 8: Accuracy of Hotz's distance metrics.

from the same distribution, against the alternative hypothesis that they come from distinct distributions.

Table 2 lists D^+ and p values when testing empirical distributions of triangle estimates vs. other estimates for each data set. We used the implementation of the K-S method in system R, where the smallest detected p value is $2.2e-16$. As we see for all data sets, the probability that the distributions are the same is negligible (below the minimum detection level). Thus, with very high probability, the alternative hypothesis – that the distributions are different – is true, which in turn means that the accuracy advantage of dynamic triangles indicated by these distributions is statistically significant in each of the environments represented by our data sets.

Since three of our four datasets rely to varying extent on PlanetLab, we wished to validate our results by focusing, in the DZ-Gnutella dataset, on the estimates between the 21 non-PlanetLab MPs we had available and the Gnutella peers. As shown in Figure 7, while accuracy of the non-PlanetLab estimates decreases somewhat, it has similar trends. In particular, the median error of the dynamic triangle estimates becomes 7.4% as compared to 13.4% for GNP, 20% for Vivaldi and 23.4% for virtual landmarks.

5.3 Early Approaches

The GNP study [26] compared its accuracy to two early approaches for distance estimation - IDMaps [7] and the Hotz's approach [10] and found that these ap-

Error	Tri	GNP	IDMaps	Hotz's U	Hotz's L	Hotz's $(L+U)/2$
Mean	15.9%	19.2%	29.5%	54.6%	60.3%	52.7%
Median	5.3%	9.6%	13.0%	8.8%	11.4%	8.5%

Table 3: Mean and median relative error of early methods.

proaches were less accurate than GNP[3]. We checked this finding on the DZ-Gnutella dataset. Following the methodology in [26], we used the same 15 landmarks selected for GNP as Hotz's landmarks so that both approaches have the same probing overhead. At the same time, because IDMaps is an asynchronous approach, we used all 100 landmarks we utilized for dynamic triangles. In the Hotz's method, for a given host-pair, we computed the highest value of the lower-bound distance estimate L and the lowest value of the upper-bound estimate U produced by all the landmarks:

$$L = \max_{i=1}^{15} |d(A, h_i) - d(B, h_i)|$$

$$U = \max_{i=1}^{15} |d(A, h_i) + d(B, h_i)|$$

where A and B are end-hosts, h_i is the ith landmark, and $d(x, y)$ is the distance between hosts x and y. We then considered values L, U, and $(L + U)/2$ as alternative distance estimates between the end-hosts.

Table 3 compares the accuracy of these estimates with those of GNP and dynamic trianges. IDMaps shows worse accuracy in terms of both median and mean error. Hotz's heuristics actually show a slight advantage over GNP in median errors but significantly lower overall average accuracy, indicating that it is vulnerable to particularly bad estimates. Indeed, as Figure 8, which depicts the CDF of the relative errors, indicates, there is a significant loss of accuracy of all three Hotz's metrics for the worst 20% of the estimates. These trends were confirmed in the other datasets: in the PL dataset, Hotz's metrics had slightly better median but worse average errors similar to the DZ-Gnutella dataset above; in the PL-Azureous and Meridian datasets both median and mean errors in the Hotz's method were worse that in GNP but the difference in the average error was much more pronounced. Furthermore, all Hotz's metrics showed lower accuracy than dynamic triangles for all the data sets and all error percentiles.

Given these findings, we do not consider IDMaps and Hotz metrics in the rest of our study.

5.4 Accuracy by Distance Ranges

Lee et al. [16] observed that coordinate approaches do not work well for estimating short distances due to large scale of triangle inequality violations. This section considers how the accuracy of our approach depends on the distance between the hosts. Figure 9 shows the median relative errors for host-pairs that fall in several real distance ranges, for dynamic triangles and existing approaches. We only show results for the DZ-Gnutella data set; other data sets had similar results.

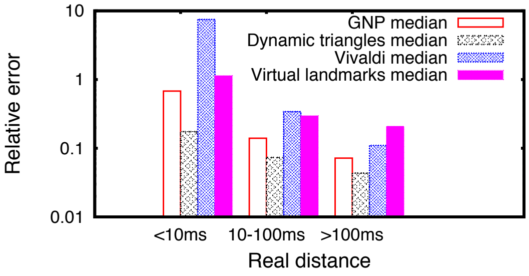


Figure 9: Relationship between estimates accuracy and inter-host distance.

This figure confirms the finding in [16], in that the errors do increase significantly for estimating shorter distances (note the logarithmic scale on Y-axis) for all methods. For example, the median relative error of distance prediction of GNP for real distance between 100 and 1000 milliseconds is just under 10%, but it increases to 68% for real distances below 10ms. (We also found that usually these estimates are biased to exceeding the real distance.) The median relative error of distance estimation of Vivaldi for the 100 to 1000ms range is only about 11% and increases to over 700% for real distances below 10 ms. Our approach also exhibits similar behavior although to much less degree. The median relative error of our approach for real distances under 10ms is only 17%, compared to GNP's 68%. We believe the reason behind this behavior is that, for short inter-host distances, our approach finds a local triangle that is close to both hosts while the coordinate-based approaches use the same host coordinates, derived from the same set of landmarks or peers, for all estimates regardless of the range. This, along with successful mitigation of triangle inequality violations (see Section 3.2) limits the error produced by our approach.

Within a limit, a somewhat higher relative error for very short distance estimation may not be significant in practice (i.e., the difference between 2ms and 4ms estimates might not be important as both indicate that the two hosts are "close"). However, the dependence of

[3] To be fair, IDMaps had a goal of asynchronously collecting the distance maps of the entire Internet and assumed many more tracers than are available in our experiments, so both [26] and ours are not apple-to-apple comparisons.

estimation quality on distance ranges shown in Figure 9 is noteworthy.

5.5 Rank Preservation

As already mentioned, in many applications, accurate ranking of hosts according to their distance to a given host is more important than the accuracy of distance estimations themselves. We now consider the two rank preservation metrics [20] mentioned in Sec. 5.1 to study the quality of dynamic triangle estimates in this respect.

The results for the common closest peers are shown in Figure 10. To produce these plots, for each data set, we first compute the percentage of overlap of the k closest hosts to a given host X according to their estimated and measured distances. The figure then plots the average overlap found for all hosts X. In the DZ-Gnutella and PL-Azureus data sets, we use, respectively, non-landmark DipZoom measurement points and PlanetLab nodes as hosts X relative to which the ranking is done. In the symmetrical PL and Meridian data sets, we use all non-landmark hosts as hosts X.

With the DZ-Gnutella dataset, we additionally separated our experiments in two groups. In one set of experiments, we only consider DipZoom measuring points as both the hosts to be ranked and the hosts relative to which the ranking is done. In the other set, we consider the ranking of all hosts (both MPs and Gnutella peers) relative to the MPs. The reason is that MPs are mostly deployed on well-connected PlanetLab nodes while Gnutella peers typically have residential connectivity. We included 220 available MPs in this study beyond the 100 landmarks. However, some MPs lacked measured distances to some hosts, and hence the number of hosts ranked relative to individual MPs may differ.

The results show that dynamic triangles have noticeably better rank preservation than the existing techniques in all data sets we considered. For example, in the DZ-Gnutella data set, we found that in the MP-only group, 125 out of 220 MPs had the same closest peer in both real and estimated distances, while this number was 113 for GNP estimations, 86 for virtual landmarks and only 8 for Vivaldi. (Thus, for $k = 1$, the graph gives 0.57 for dynamic triangles, 0.52 for GNP, 0.39 for VL, and 0.036 for Vivaldi.) Considering the top-10 closest MPs, on average, 7.29 out of 10 closest peers predicted by dynamic triangles were among true top-10 closest peers according to real distances; the same was true for 6.06 out of 10 closest peers for GNP, 4.76 out of 10 for virtual landmarks, and 2.45 out for 10 for Vivaldi. In the second group, where ranked hosts are dominated by residential nodes, all methods showed accuracy degradation, especially for very low k, but dynamic triangles still retained an edge. We finally note that the rank preservation of GNP according to the common closest peers metric in our all-hosts experiment is similar to that reported in the GNP study [26].

Method	DZ-Gnutella (MP-only)	DZ-Gnutella (All hosts)	PL	PL-AZ	Meridian
TRI	5.83%	11.19%	5%	13.9%	21%
GNP	9.35%	12.72%	8.4%	15.7%	24%
VL	17.52%	28.3%	10.2%	19.2%	35.2%
Vivaldi	19.28%	24.2%	5.5%	19.3%	24.9%

Table 4: Relative rank loss

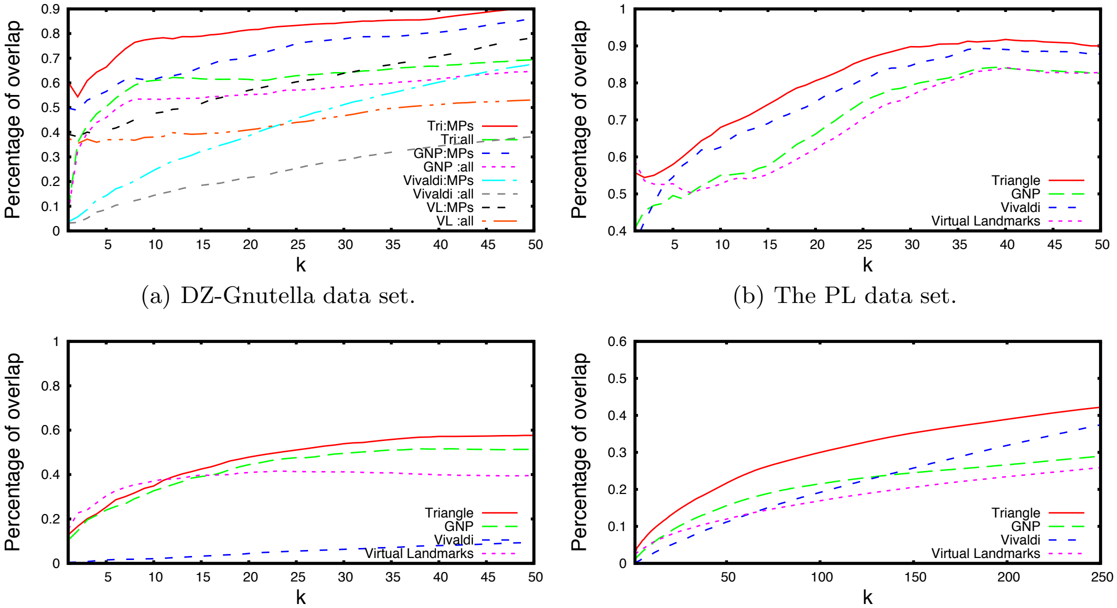
Turning to the relative rank loss, Table 4 shows the average value of this metric of different approaches in our data sets. In each data set, we computed this metric relative to each available host and took the overall average, for each approach. As seen from this table, while the relative rank loss of the dynamic triangles varies in different data sets, it is again always better than in the existing approaches. Lua et al. [20] previously compared the relative rank loss of Vivaldi and the virtual landmarks and they found their mean relative rank loss to be mostly between 15 and 20%.

In summary, our rank preservation study shows that, in addition to lower relative error, dynamic triangles achieve better ranking accuracy than existing approaches, in many cases by a significant margin. Within the dynamic triangles approach, as seen from results from the MP-only experiments on the DZ-Gnutella data set as well as from the PL data set, the top-1 selection works well when ranking well-connected hosts, picking the true best host more than half the time. At the same time, the results from the from all-hosts in the DZ-Gnutella data set and from PL-Azureous data set indicate that top-1 selection performs much worse for residential peers. However, the ranking accuracy of residential hosts increases rapidly towards top-10 selection. This behavior bodes well for peer-to-peer applications, where top-1 selection is typically limited to well-connected super-peers while top-k selection often includes other residential peers.

6. DESIGN ISSUES

We now examine the effect of several design and configuration issues on the accuracy of the dynamic triangle estimates. In particular, we consider the significance of dynamic triangle selection (vs. using a common big triangle for all host-pairs), the effect of the number of candidate landmarks on the accuracy of the approach, the impact of triangle inequality violations, the impact of using the estimated host-to-landmark distances in triangle selection, and the effect of our heuristic for choosing the initial value of the threshold described in Section 3. We used the DZ-Gnutella data set for all the experiments in this section.

6.1 Impact of Dynamic Triangle Selection



(a) DZ-Gnutella data set.

(b) The PL data set.

(c) PL-Azureus data set.

(d) Meridian King data set.

Figure 10: The overlap between estimated and real top-k closest peers.

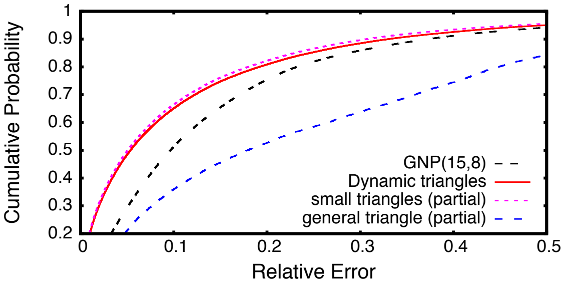


Figure 11: The significance of dynamic triangle selection.

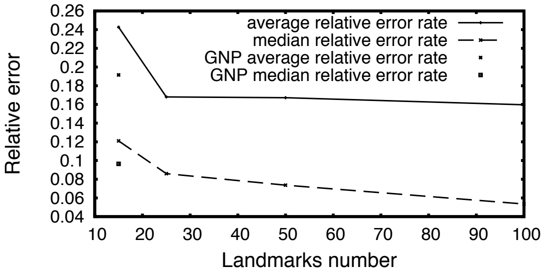


Figure 12: The effect of the number of landmark candidates on accuracy.

We begin by considering the significance of dynamically selecting a landmark triangle for a given host-pair. Figure 11 separates the accuracy of distance estimates for the host-pairs that used dynamically selected small triangles from the host-pairs for which a suitable small triangle could not be identified and thus the general static triangle had to be used. The figure also reproduces the curves for the combined accuracy in our approach and for the accuracy of of GNP (15,8). The figure shows that the accuracy suffers significantly when no suitable small triangle is found. Fortunately, with 100 candidate landmarks, very few host pairs – only 10780 host-pairs out of a quarter million total – had to resort to the general triangle so the overall accuracy was not affected. However, the result of Figure 11 confirms our intuition from Section 4.1 that dynamic selection of an appropriate landmark triangle is essential to the accuracy of our approach.

6.2 Number of Landmark Candidates

Our next question is how many landmark candidates are needed in our approach. To address this question, we have considered the accuracy of our approach using 15, 25, 50 and 100 landmark candidates. In each experiment, we select a given number of landmark candidates from the original 100 landmarks in the same way we chose the original 100 candidates from all suitable hosts. Specifically, we partition the 100 candidate landmarks into 15, 25 and 50 clusters (using k-mean clustering based on measured inter-host distances) and choose hosts closest to each cluster center as the landmark candidates.

Figure 12 presents the effect of the number of landmark candidates on the accuracy of our approach. As

Num. of violations	0	1	2	3
Prevalence	25.7%	34.3%	22.1%	17.8%
Median rel. error	4.2%	5.2%	5.7%	6.7%

Table 5: the degree of triangle inequality violations and its effect on accuracy.

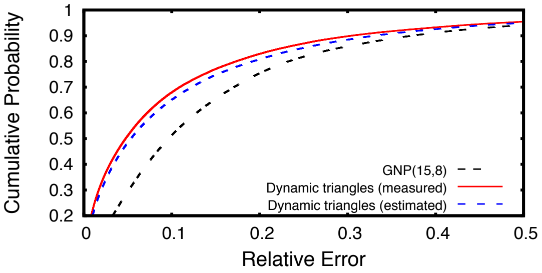


Figure 13: Estimated vs. measured triangle selection.

we can see, both the average and median relative error rate decrease as the number of candidate landmarks increases from 15 to 100, as host-pairs are more likely to find an acceptable small triangle. However, the decrease is most significant for low numbers (15-25) of landmarks. This result is encouraging because it suggests that high-quality distance estimates can be achieved by maintaining a relatively limited set of landmarks. Certainly 100 landmark candidates used in our study is adequate, and fewer numbers can be sufficient depending on the application.

6.3 Triangle Inequality Violations

Turning to the impact of triangle inequality violations, their effect on accuracy of our approach is shown in Table5. For a given pair of end-hosts and the selected dynamic triangle, the number of triangle violations in our approach could range from 0 to 3 (Section 3.2), and most host-pairs had at least some violations. As one would expect, we found a direct dependence between the number of violations and the inaccuracy of the distance estimation for a given host-pair. However, the loss of accuracy is modest, ranging from 4.2% for no violations to 6.7% for three violations. We conclude that our approach successfully mitigates the effect of triangle inequality violations.

6.4 Using Estimated Distances in Triangle Selection

Our approach selects the best landmark triangle for a given host-pair based on the estimated distance between these hosts and all landmark candidates. To consider how much accuracy is lost due to using estimated distances in triangle selection, Figure 13 compares the accuracy of the estimates produced by our triangle selection with the selection using measured distances. Recall from Section 4.4 that the latter is an idealistic method due to its prohibitive probing traffic. Fortunately, as the figure shows, the use of estimated distances in triangle selection does not lead to considerable accuracy loss. The selection based on measured distances has the median error of 4.7% compared to 5.3% in the selection using estimated distances.

7. APPLICATIONS

While accuracy of distance estimates is important, their true utility transpires in the context of specific applications using them. In this section, we evaluate our approach when used for two representative types of applications - host clustering and content delivery networks.

7.1 Host Clustering

An important application of distance estimation is host clustering, which is used in a large variety of contexts, including peer-to-peer systems, scalable network monitoring, and content dissemination. Pair-wise distance measurements between nodes would ideally drive clustering algorithms but are often unavailable, either due to the system size (since the number of measurements grows quadratically with the number of nodes), or because of the lack of control over the nodes. In these cases, distance estimates provide a viable alternative, and have been used, e.g., by Chen et al. in the context of network monitoring [2]. We evaluate distance estimation techniques with respect to clustering by considering how clusters formed using estimates provided by the corresponding approach deviate from clusters formed using true measured distances.

We again used DZ-Gnutella dataset for this experiment. We selected 99 non-landmark MPs for which we have the complete matrix of real (i.e., measured) inter-MP distances and used the k-mean clustering algorithm to group them into five clusters. To compare the real-distance clustering with a given estimated-distance clustering, we pair up clusters from each clustering that have the largest number of nodes in common; we then compute the total number of mismatched nodes across all cluster pairs. We consider two dynamic triangle configurations - with 100 and 25 landmark candidates. Note that Vivaldi is intended for peer-to-peer applications; it would not be used for pure clustering and is included in this study for completeness.

Table 6 shows the performance results. We see that dynamic triangle estimates produce the highest quality clusters, with only 5 out 99 nodes mismatched. This followed by GNP, but dynamic triangle incurred drastically lower computational cost to produce all the distance estimates than GNP. Virtual landmarks had the lowest computational overhead (we measured it with a

Method	Computation time	mismatched nodes	Median rel error	Network probings
TRI 100	2 sec	5	2.2%	4546
TRI 25	1 sec	5	4.5%	1187
GNP	6 min	11	7.3%	1485
VL	< 1 sec	20	29%	1980
Vivaldi	NA	17	18.3%	9900

Table 6: Host clustering performance

second granularity and their overhead was below our ability to measure).

To measure the probing overhead, we counted the actual number of probes required in each approach to produce all the distance estimates. As explained earlier, we only count the number of probes between the landmarks and hosts, because the periodic probes among the landmarks are negligible. Dynamic triangles with 100 landmarks generate three times more probes than GNP; the reason is that, although we use fewer probes for each estimate, the same host may probe different triangles for different estimates while a host in GNP always reuses its probes.

Interestingly, our approach produced identical clusters using 25 and 100 landmarks, although going from 100 to 25 landmarks doubled the median relative error of inter-host distance estimates. With 25 probes, dynamic triangles actually generated slightly lower probing traffic than GNP. This shows that in some scenarios, one can reduce the costs of the estimates without sacrificing their value. Systematic ways to exploit this potential for optimization is an interesting question for future work.

Overall, while these results are based only on one set of hosts and should be viewed as preliminary, dynamic triangle estimations produced better-quality clusters than GNP in this case.

7.2 CDN Server Selection

Another appealing application of network distance estimation is to use it for server selection in a content delivery network (CDN). While even efficient estimates would be too heavy-weight for online server selection in the context of delivering small HTTP objects, CDNs are increasingly used to deliver large files (such as software packages and multimedia files) and long-running streaming media. With dynamic triangles being able to estimate 10,000 distances in single seconds (see Table 6) this clearly makes it feasible to use online distance estimates in server selection in these contexts.

Recently, Szymaniak et al. [30] studied the application of GNP estimates to select among ten servers in the Google content delivery network and found them quite effective. We consider the application of dynamic triangle estimates to a large-scale CDN, using Akamai with its thousands of servers as a target environment. In this section, we first compare the performance of our estimates in this application with existing approaches using a simulation study on a data set employed in the original Meridian study [35]. We then report the results from a live Internet experiment that measures the quality of server selection in the Akamai platform by the dynamic triangles and GNP.

7.2.1 Simulation Study

We now compare the quality of server selection using dynamic triangle estimates with several alternative techniques that have been proposed for this purpose – GNP, Vivaldi, and Meridian. We use the Meridian King data set and follow the general methodology of Meridian study in this experiment. In particular, we are interested in the general quality of server selection promised by each approach, thus assuming a specially modified Web client. We use the absolute value of the difference in distance from a client to its true closest server and from the same client to the server selected by a given approach, as the measure of quality of the approach.

The Meridian King data set [23] contains the distance between 2500 local DNS servers collected using King method. We randomly pick 500 nodes as CDN clients and apply different techniques to select the closest server among the remaining 2000 nodes. In the case of dynamic triangles, we use one-sided triangle selection as discussed in Section 4.3.

The original median absolute difference in our approach is 12.3ms, which is similar to the results using (15,8)-GNP and Vivaldi in the Meridian study [35] and much higher than the Meridian system, i.e., about 1.3ms observed from Figure 8 in [35]. However, Meridian's accuracy comes with a high probing cost. According to [35], the average bandwidth cost for each closest server discovery is 10.4KBytes with a probe packet size of 50 bytes. This translates to 104 probes if we assume each probe generates two packets. As mentioned in the Meridian study, with this "budget" of active probes, other techniques could improve their selection quality by using distance estimates only to predict top-k candidate closest servers and actively probing all these k servers to arrive at the final selection.

Figure 14 compares Meridian server selection with that of GNP, Vivaldi and our approach with this optimization, for different sizes k of the candidate server set. It shows that the median absolute difference drops sharply as k increases for all three approaches using distance estimates, especially in our case. For $k = 10$, the quality of our approach reaches the Meridian's, while our system only uses 6+10=16 probes and Meridian uses over 100 probes on average. For higher values of k, the quality of our approach improves beyond the Meridian's level while the total probing cost of our approach

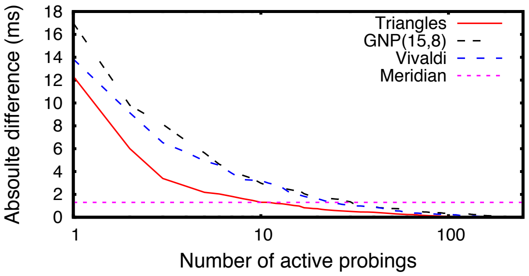


Figure 14: The absolute difference for closest servers selection with active probings.

is still lower than Meridian.

The above cost analysis does not include maintenance costs, but these costs are negligible (compared to the cost of content delivery to clients) and similar in both systems. For dynamic triangles, the full complement of maintenance measurements includes pair-wise probes between all landmarks as well as between landmarks and servers. With 2000 servers and 100 landmarks, this translates into $2000 * 100 + 100 * 99/2 = 204950$ probes. For Meridian, each server probes servers in its rings. In the experiment of Figure 14, a server has nine rings and up to 16 servers in each ring [35], for roughly $2000 * (9 * 16) = 288000$ probes. Allowing for some probe reuse and some rings having fewer servers, this results in roughly similar maintenance costs.

7.2.2 A Live Internet Experiment

To compare our approach with GNP in a realistic setting, we implemented an AJAX application that, when loaded by a client, performs server selection among Akamai edge servers using dynamic triangles, GNP, and Akamai itself, and reports to us the latency from the client to each of the three servers. Following [30], we used GNP(7,6) configuration in this study. We picked a CNAME (a1694.g.akamai.net, utilized by pcworld.com) which we found is mapped by Akamai to a large number of edge servers, 979. We request a bogus URL from the selected server in all cases, receiving a short “Bad Request” response. We verified through tcpdumps on our own client that downloading this response involves two round-trip times (RTTs) between the client and server.

We have maintained about 90 landmark servers on the PlanetLab platform (we attempt to maintain up to 104 but the actual number varies due to the instability of PlanetLab nodes). We keep the complete distance matrix between all the landmarks as well as between each landmark and each of the 979 Akamai server. We select (through clustering as described in Section 5.1) three of these landmarks as the general big triangle for our approach, and seven landmarks for GNP(7,6). Given the instability of PlanetLab nodes, we select, for each of the above landmarks, a few nearby landmarks as backups.

We implemented three Web pages, one for each server selection. The Akamai page includes Javascript that simply accesses a bogus URL with the above CNAME, thus following the Akamai server selection, and reports the response time to our server. To factor out DNS resolution time, we perform the download twice and report the second measurement.

The triangles page[4] embeds three images with URLs pointing to the landmarks at the vertexes of the big triangle[5]. As the client establishes the TCP connections to these landmarks, they passively measure RTT to the client, select a small triangle based on these measurements[6], and then respond to the client with an HTTP redirect to three URLs pointing to the vertexes of the small triangle. The latter landmarks similarly measure RTT to the client, estimate the distance from the client to all 979 Akamai servers and select the best one. Finally, one of the landmarks returns a redirect to the selected server using its raw IP address (and a bogus URL) while the other two landmarks return an empty image. The client follows the redirect and reports the download time to our server.

The GNP page[7] is implemented similarly but embeds seven images (to measure RTT to the seven landmarks) and involves one round of landmark communications instead of two.

We asked a commercial company to embed zero-sizes iframes with the above three pages, and we also embedded them into our own Web pages. We have collected 24,079 measurements from 2,926 distinct client IP addresses, representing 47 US states and 43 foreign countries according to the GeoIP database from MaxMind [22].

Figure 15 shows the CDF of the measured RTT from clients to Akamai servers selected by the three methods, and Table 7 lists their median and mean values. These results show the performance of the dynamic triangles is significantly better than GNP(7,6), with almost a factor of two reduction in median client-to-server RTT (32ms vs 62.4ms). It is well established that latency directly affects TCP throughput (see, e.g., [27]) and hence this latency reduction would translate directly into the reduction in web page download time as experienced by the user. In fact, the server selection quality with our simple technique is fairly close to that of Akamai itself, despite Akamai’s extensive Internet-wide measure-

[4] See http://haddock.case.edu:8000/triangle.

[5] This is a slight simplification. In fact, these URLs point to our portal server which uses HTTP redirect to to send the client to the landmarks. This level of indirection allows us to replace failed landmarks with their backup dynamically.

[6] This and other actions requiring central computation are done through a back-end server.

[7] See http://haddock.case.edu:8000/gnp.

Method	Median RTT (ms)	Mean RTT(ms)
Akamai	24.8	52.4
Triangle	32	62.4
(7,6) GNP	62.4	88.5

Table 7: Performance of closest CDN server selection for web clients.

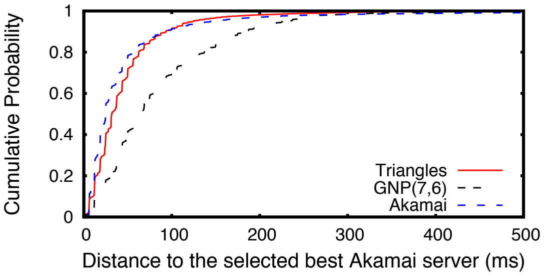


Figure 15: The quality of CDN server selection in a live deployment.

ments and network topology expertise that go into its server selection decisions [17, 18]. By adding a few active probes to top-k servers as described in Section 7.2.1, we hope to be able to close this gap in the future.

At the same time, dynamic triangles require 6 probes for each server selection vs. 7 probes needed by GNP(7.6). These 6 probes do come in two sequential rounds while GNP sends all its probes in parallel. However, for large downloads we are targeting, finding a better server is likely to be worth this small initial delay.

8. CONCLUSION

This paper presents a simple and efficient approach for estimating the network distance between Internet hosts. Our approach uses a set of hosts that act as landmark candidates, and dynamically selects a triangle of landmarks for a given pair of hosts that are likely to produce high-accuracy distance estimates. Once the landmark triangle is selected, the estimate is computed from a simple trigonometrical calculation. Through extensive testing we showed that our approach compares favorably with representative existing methods for distance estimation, both from the general accuracy and overhead perspective and in the context of some specific applications.

We are currently implementing our approach in the BitTorrent tracker and Vuze client. As future work, we would like to investigate some optimizations to our approach we mentioned throughout the paper, most notably, accounting for triangle shape during dynamic triangle selection and using a replacing the triangle with a single landmark when it is particularly close to one of the end hosts. We would also like to combine our approach with some orthogonal techniques that have been proposed to enhance distance estimation and measurement [15, 16].

Appendix A: Trigonometric Derivations

Assume we have 3 landmarks $L1$,$L2$ and $L3$ and two points A and B. We also know the distance between landmarks and their distance to points A and B, and we need to find the distance between A and B. In the formulas below, we use $\overline{P1,P2}$ to denote the edge between points $P1$ and $P2$ and $|\overline{P1,P2}|$ to denote its length. We use $\angle(P1,P2,P3)$ to denote the angle between edges $\overline{P1,P2}$ and $\overline{P2,P3}$.

Points A and B each can have two possible positions, A', A'', B' and B'' as illustrated in Figure 1 and 2. Since we know the length of edges $\overline{A,L1}$, $\overline{A,L2}$, and $\overline{L1,L2}$, we can calculate the cosine of angle $\angle(A,L1,L2)$ using the following triangle function, regardless of whether A is positioned at A' or A'':

$$\cos\angle(A,L1,L2) = \frac{|\overline{A,L1}|^2 + |\overline{A,L2}|^2 - |\overline{L1,L2}|^2}{2*|\overline{A,L1}|*|\overline{A,L2}|} \qquad (1)$$

We can obtain the cosine of angle $\angle(B,L1,L2)$ using the same method. As the next step, we would like to find the cosine of $\angle(A,L1,B)$. However, depending on whether or not points A and B are on the same side on line $\overline{L1,L2}$, angle $\angle(A,L1,B)$ can be the sum of two angles $\angle(A,L1,L2)$ and $\angle(B,L1,L2)$, or the difference between these angles. Thus, its cosine will be:

$$\begin{aligned} \cos\angle(A,L1,B) &= \cos(\angle(A,L1,L2) \pm \angle(B,L1,L2)) \\ &= \cos\angle(A,L1,L2) * \cos\angle(B,L1,L2) \\ &\mp \sin\angle(A,L1,L2) * \sin\angle(B,L1,L2) \quad (2) \end{aligned}$$

where we use the upper arithmetic operation if A and B are on the same side on line $\overline{L1,L2}$ and the lower arithmetic operation otherwise.

In order to determine whether A and B are on the same side of $\overline{L1,L2}$, we utilize point $L3$ as a witness point. First, we determine if A and $L3$ are on the same side of line $\overline{L1,L2}$ or not. To this end, we obtain the cosines of angles $\angle(A,L1,L2)$ and $\angle(L3,L1,L2)$ using the equation similar to 1, and use these cosine values to compute the two possible cosine values of $\angle(A,L1,L3)$ using the equation similar to 2. We also obtain the cosine of $\angle(A,L1,L3)$ directly from the edges, using equation similar to 1 and compare it to the two values found from 2. Depending on which of these two values are closer, we conclude whether A and $L3$ are on the same side of line $\overline{L1,L2}$ or not. We repeat the same method to determine whether or not points B and $L3$ are on the same or opposite sides of $\overline{L1,L2}$. We then can answer our question on the post ion of points A and B relative to $\overline{L1,L2}$: if both A and B are on the same side with $L3$, or both are on the opposite side from $L3$, then both A and B are on the same side of $\overline{L1,L2}$. Otherwise, they will be on the opposite sides of $\overline{L1,L2}$.

Finally, we can use the above conclusion to chose between the two alternative values of $\cos\angle(A, L1, B)$ and, knowing now this cosine value, calculate the estimated distance between A and B using the following equation 3:

$$|\overline{A,B}|^2 = |\overline{A,L1}|^2 + |\overline{B,L1}|^2 + 2|\overline{A,L1}||\overline{B,L1}|\cos\angle(A, L1, B) \tag{3}$$

9. REFERENCES

[1] Akamai Technologies. http://www.akamai.com/html/technology/index.html.

[2] Y. Chen, K. H. Lim, R. H. Katz, and C. Overton. On the stability of network distance estimation. *Perf. Eval. Rev.*, 30(2):21–30, 2002.

[3] M. Costa, M. Castro, A. Rowstron, and P. Key. PIC: Practical Internet coordinates for distance estimation. In *ICDCS*, 2004.

[4] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. In *SIGCOMM*, 2004.

[5] DipZoom. http://dipzoom.case.edu.

[6] http://vorlon.case.edu/∼zxw20/distance/.

[7] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: a global Internet host distance estimation service. *IEEE/ACM Trans. Netw.*, 9(5):525–540, 2001.

[8] GNP. www.cs.rice.edu/∼eugeneng/research/gnp.

[9] K. Gummadi, S. Saroiu, and S. Gribble. King: Estimating latency between arbitrary internet end hosts. In *SIGCOMM Internet Measurement Workshop*, 2002.

[10] S. M. Hotz. Routing information organization to support scalable interdomain routing with heterogeneous path requirements. Ph.d. thesis, University of Southern California, 1994.

[11] F. James. *Statistical Methods in Experimental Physics (2nd Edition)*. World Scientific, 2006.

[12] J. Jung, B. Krishnamurthy, and M. Rabinovich. Flash crowds and denial of service attacks: characterization and implications for CDNs and Web sites. In *Proc. 8th Int. Conf. on WWW*, 2002.

[13] B. Krishnamurthy and J. Wang. On network-aware clustering of web clients. In *SIGCOMM*, 2000.

[14] J. Ledlie, P. Gardner, and M. Seltzer. Network coordinates in the wild. In *NSDI '07*.

[15] J. Ledlie, P. Pietzuch, and M. Seltzer. Stable and accurate network coordinates. In *ICDCS '06*.

[16] S. Lee, Z.-L. Zhang, S. Sahu, and D. Saha. On suitability of euclidean embedding of internet hosts. *Perf. Eval. Rev.*, 2006.

[17] F. T. Leighton, R. Sundaram, and M. Levine. Method for generating a network map. Akamai Technologies, Inc. US Patent No.: 7,251,688.

[18] M. Levine, R. Kleinberg, and A. Soviani. Method for extending a network map. Akamai Technologies, Inc. US Patent No.: 7,028,083.

[19] H. Lim, J. C. Hou, and C.-H. Choi. Constructing internet coordinate system based on delay measurement. *IEEE/ACM Trans. Netw.*, 13(3):513–525, 2005.

[20] E. K. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft. On the accuracy of embeddings for Internet coordinate systems. In *IMC'05*.

[21] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee. Triangle inequality and routing policy violations in the internet. In *PAM*, pages 45–54, 2009.

[22] MaxMind. http://www.maxmind.com.

[23] Meridian: Lightweight positioning. http://www.cs.cornell.edu/People/egs/meridian/.

[24] Network coordinate research at harvard. http://www.eecs.harvard.edu/∼syrah/nc/.

[25] T. S. E. Ng and H. Zhang. A network positioning system for the internet. In *ATEC'04*.

[26] T. S. E. Ng and H. Zhang. Predicting internet network distance with coordinates-based approaches. In *INFOCOM*, 2002.

[27] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose. Modeling tcp reno performance: a simple model and its empirical validation. *IEEE/ACM Trans. Netw.*, 8(2):133–145, 2000.

[28] M. Pias, J. Crowcroft, S. Wilbur, S. Bhatti, and T. Harris. Lighthouses for scalable distributed location. In *IPTPS*, 2003.

[29] Y. Shavitt and T. Tankel. Big-bang simulation for embedding network distances in euclidean space. *IEEE/ACM Trans. Netw.*, 12(6):993–1006, 2004.

[30] M. Szymaniak, D. Presotto, G. Pierre, and M. van Steen. Practical large-scale latency estimation. *Comput. Netw.*, 52(7):1343–1364, 2008.

[31] L. Tang and M. Crovella. Virtual landmarks for the Internet. In *IMC'03*.

[32] Z. Wen. *Simplifying End-to-End Mesurement on the Internet*. PhD thesis, Case Western Reserve University, 2009.

[33] Z. Wen and M. Rabinovich. Network distance estimation with dynamic landmark triangles (Poster paper). In *SIGMETRICS*, 2008.

[34] Z. Wen, S. Triukose, and M. Rabinovich. Facilitating focused internet measurements. In *SIGMETRICS*, 2007.

[35] B. Wong, A. Slivkins, and E. G. Sirer. Meridian: a lightweight network location service without virtual coordinates. In *SIGCOMM05*, 2005.

[36] B. Zhang, T. S. E. Ng, A. Nandi, R. H. Riedi, P. Druschel, and G. Wang. Measurement based analysis, modeling, and synthesis of the internet delay space. In *Internet Measurement Conference*, pages 85–98, 2006.

[37] H. Zheng, E. K. Lua, M. Pias, and T. G. Griffin. Internet routing policies and round-trip-times. In *PAM*, pages 236–250, 2005.