Client Clustering for Traffic and Location Estimation

Lisa Amini[1,2] and Henning Schulzrinne[2]
IBM Research[1] and Columbia University[2]

Abstract

Resource management mechanisms for large-scale, globally distributed network services need to assign groups of clients to servers according to network location and expected load generated by these clients. Current proposals address network location and traffic modeling separately. In this paper, we develop a novel clustering technique that addresses both network proximity and traffic modeling. Our approach combines techniques from network-aware clustering, location inference, and spatial analysis. We conduct a large, measurement-based study to identify and evaluate Web traffic clusters. Our study links millions of Web transactions collected from two world-wide sporting event websites, with millions of network delay measurements to thousands of Internet address clusters. Because our techniques are equally applicable to other traffic types, we expect they will be useful in a variety of wide-area distributed computing optimizations, and Internet modeling and simulation scenarios.

1. Introduction

Large-scale Internet applications are often deployed on widely distributed servers to improve performance and availability. In such environments, server selection mechanisms are required to direct clients to a network proximal server [6], [17]. Clients are grouped, or clustered, according to network location since tracking network delays and making server assignments on a per-client basis would severely limit scalability. Server utilization may be monitored and factored into the selection process, but it is more difficult to project the expected load a set of clients may impose. Without this projection, certain servers may be overloaded during traffic surges. Clients may be redirected to alleviate overload, but since feedback is delayed, assignments may oscillate, resulting in widely varying response times for affected clients.

We define a traffic cluster (TC) as a group of IP addresses that can be treated as a unit, with respect to application level traffic patterns and network location. Using TCs, server selection algorithms could improve resource management by also considering the expected load client sets may impose. Additionally, technologies for distinguishing between legitimate surges and denial of service attacks could use historical data to better assess the likelihood that one or more TCs in given locations could legitimately produce a given workload.

Another example arises in intelligent route control mechanisms [18], which enable multi-homed Internet data centers to adapt to network conditions by switching among different network service provider connections. Currently, a route shift is predicated on the expected network delay of a cluster through a given connection, without attempting to project the effect the shift might have on link utilization. Using TCs, these mechanisms could better evaluate whether a given change might result in overload or increased peak utilization charges.

The applicability of traffic cluster analysis is not limited to uses in intelligent server and network management. The ability to represent network structure with simple, yet realistic models is the key motivating factor in mapping the Internet. Network structure is not limited to physical connections; the logical sessions formed over physical connections also play a key role. TCs represent both the logical (application level traffic) and physical (topological and geographic proximity) properties of Internet structure. TC statistical properties, including inter- and intra- cluster delays, cluster traffic intensity, and cluster density, offer the potential of modeling via point processes. Such models could, in turn, enable new simulation techniques.

In this paper, we develop a methodology to identify and characterize Web traffic clusters. Our contributions are as follows.

- We perform an objective, quantitative evaluation of network delay estimation techniques, based on empirical data from full mesh probing of 113 widely distributed servers, and validate a specific technique for use in identifying Web traffic clusters.
- We propose a novel traffic clustering technique that combines network-aware clustering, location-aware clustering and new applications of Voronoi tessellations.
- We evaluate the identified TCs in terms of network proximity and ability to represent significant traffic sources. Our evaluation is done in the context of millions of Web cache transactions from the official Web sites for two internationally popular sporting events.
- We demonstrate an effective application of the identified TCs to intelligent server selection under real-world conditions and show our technique outperforms competitive schemes by a factor of more than two.

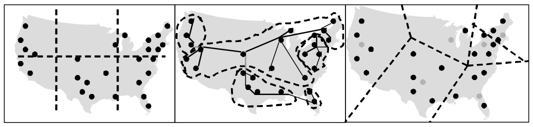

a) binned b) domain c) nearest station

Figure 1: Clustering techniques. a) Address coordinates are binned to create clusters, b) addresses with a common administrative domain are assigned to a single cluster, c) spatial partitioning of the plane in which a number of stations (depicted in gray) are designated and each host (depicted in black) is assigned to the nearest station.

Our evaluation is done in the context of Web traffic, but our techniques are applicable to any large-scale, distributed application in which it is important to identify groups of clients that impose significant portions of the overall workload and are network proximal.

The remainder of this paper is organized as follows. In the next section we provide background information, and introduce our clustering algorithm in Section 3. We describe our data collection methodology in Section 4. We detail the results of our analysis in Section 5 and offer conclusions in Section 6.

2. BACKGROUND

Our clustering algorithm leverages recently developed techniques to estimate network delay between IP hosts. In this section, we briefly review prior work in network delay inference and client clustering. We use the terms network delay and distance interchangeably to refer to the minimum round trip time (RTT) between an IP address pair. A comprehensive evaluation of delay and other network distance metrics is provided in [11].

2.1 Inferring Internet Distance

Several network delay estimation schemes have been proposed, which we broadly classify as using either i) *via* or ii) *geometric* heuristics.

Under the *via* heuristic, the distance between an arbitrary pair of Internet hosts, $\mathcal{H}_1$, $\mathcal{H}_2$, is based on direct measurements between a number, *N*, of fixed locations, which we refer to as probe stations. We identify probe station *i*, as P_i, where $i \in \{1,2,\ldots,N\}$. The network distance between $\mathcal{H}_{1,}$ and $\mathcal{H}_2$, d($\mathcal{H}_2$, $\mathcal{H}_2$), is estimated as the sum of the distances between $\mathcal{H}_1$ and its nearest probe station P_i, $\mathcal{H}_2$ and its nearest probe station $P_{j,}$, and the distance between P_i and $P_{j,}$. IDMaps [8] and King [9] are examples of systems using the *via* heuristic.

Geometric heuristics were originally proposed for inter-domain routing by Hotz [10]. Hotz defined the location of a given host, $\mathcal{H}_i$, in Internet space as an *N*-tuple:

$L(\mathcal{H}_i)=<\ d(\mathcal{H}_i, P_1), d(\mathcal{H}_i, P_2), d(\mathcal{H}_i, P_3), \ldots, d(\mathcal{H}_i, P_N), >$

where each element represents the distance between $\mathcal{H}_i$ and the probe station for the corresponding position in the vector. This *N*-tuple is often referred to as a coordinates vector (CV). If the triangle inequality were to hold for Internet distances, the distance between a pair of arbitrary hosts $\mathcal{H}_1$, $\mathcal{H}_2$, would be bounded from below by $\mathcal{L}=\mathrm{MAX}_i\,(|d(\mathcal{H}_1, P_i) - d(\mathcal{H}_2, P_i)|)$, and from above by $\mathcal{U}=\mathrm{MIN}_i\,(d(\mathcal{H}_1, P_i) + d(\mathcal{H}_2, P_i))$, where $i \in \{1,2,\ldots,N\}$. Hotz provided the rationale for this formulation and showed that the average, $(\mathcal{L}+\mathcal{U})/2$, generally produces better results than $\mathcal{L}$ or $\mathcal{U}$. In later sections, we refer to $\mathcal{L}$ as MaxD, $\mathcal{U}$ as MinS, and $(\mathcal{L}+\mathcal{U})/2$ as LU2.

Distance estimation using geometric heuristics has also been cast as a global minimization problem [15]. In this case, distance estimation is based on *absolute* coordinates, as opposed to the *relative* coordinates proposed by Hotz. This method assigns an absolute CV to each probe station, or landmark. The goal is to assign CVs that minimize the overall error between measured and computed distances between landmarks. Once the absolute CVs are computed, arbitrary hosts can measure their distance to the *N* landmarks, and derive their CVs based on measurements to the landmarks and the CVs assigned to the landmarks. We refer to this distance estimation method as AbCo in later sections.

In Section 4.1, we describe our comprehensive evaluation of these inference techniques, and select the technique best suited for our environment.

2.2 Clustering Techniques

Clustering techniques attempt to group sets of hosts based on network or geographic proximity, or administrative domain. For example, hosts can be grouped, or binned, according to the network distance measurements described in Section 2.1. One can achieve a coarse-grain partitioning by dividing each CV element by a common divisor. Under this method, as depicted in Figure 1a, hosts are grouped, but the groups may not represent clusters. For example, a single natural cluster may be unnecessarily split, and multiple natural clusters may be inappropriately combined.

Alternatively, client addresses may be grouped according to their administrative domain. For example, clients with the same first 24 bits in their IP address may be grouped. We use the common convention of referring to the portion of the address used for grouping as a *prefix*, and to IP addresses grouped by their 24-bit prefix as /24 clusters. An advantage of grouping by address prefix is that it does not require network delay measurements. The disadvantage is that /24 clustering results in a very large number of clusters, each generating small amounts of traffic.

A recent study [14], demonstrated that the variable length prefixes stored in Border Gateway Protocol (BGP) router tables provide a better means of identifying the administrative domains of IP addresses than fixed prefix lengths. We refer to the clusters identified in this manner as BGP clusters. As shown in Figure 1b, BGP clusters may span a wide area, such as those of transport providers. Likewise, clients in close geographic proximity, but separate domains, are not grouped.

Finally, consider how telecommunications networks are actually deployed. Network providers distribute stations (i.e., points of presence) regionally or spanning multiple geographic regions. Customers connect to nearby stations, and in turn, may provide connections to their customers or employees. The connection density in a region tends to follow population density. As depicted in Figure 1c, by associating each host to its nearest station, the plane can be partitioned into disjoint, convex polygons. This spatial partitioning is referred to as a Voronoi diagram, and each unit is referred to as a Voronoi cell.

3. TRAFFIC CLUSTERING

In this paper, we propose a simplified representation of traffic clusters wherein each cluster is identified as a Voronoi cell. In our mapping, the cell station does not represent a routing concentration point, but instead, a probabilistic location for the hosts assigned to the cell, or cluster. Minimizing the maximum intra-cell delay for a given number of stations or minimizing the number of stations for a given maximum intra-cell delay are NP-complete problems. Thus, identification and validation of heuristics for determining an effective number of stations, locations for stations, and host-to-station assignments are key contributions of our work.

Intuitively, clusters could be formed by creating a TC for each /24 cluster, and then iteratively merging the cluster with the fewest client requests to a particular application into its nearest neighbor. This creates larger clusters (in terms of traffic volume), which can improve traffic prediction capabilities. Alternatively, the merge criteria could be location-based – i.e., the closest pair of clusters is merged at each iteration.

The issue in each case is that optimization is along a single dimension (traffic volume or network proximity). Since simultaneous optimization over both dimensions is not computationally feasible, we chose to leverage human population-based clustering. More specifically, we use heuristics to identify a set of seed locations (locations likely to represent human population or Web traffic centers), and then refine the coordinates used to represent those centers with a technique based on iterative partitioning [13].

3.1 Initial Seed Station Selection

To identify network locations likely to represent Web traffic centers, we used an extensive repository of network probe data and website access logs. These data sets are detailed in Section 4.2.

Our seed selection algorithm begins by computing absolute CVs (as described in Section 2.1.2) for all probed /24 clusters, and compiling a list of CVs at which one or more /24 clusters were identified. Each CV in this list represents a potential seed location. Each probed /24 cluster is assigned to the nearest potential seed. To avoid over representing any single metropolitan area, we round each element of the CVs in this list to the nearest multiple of 5 ms. We chose 5 ms based on typical latencies [16] observed within a single metropolitan area.

For each potential seed location, we compute a weight, which is the total traffic observed from the /24 clusters assigned to that potential seed. The potential seed locations are ranked according to this traffic-based weight; the CVs of highest-ranking stations serve as seed locations.

We describe our technique for selecting the number of seed stations in Section 5.1.1. Next, we describe how this list of seed stations is refined into a final partitioning.

3.2 Refining Cluster Station Coordinates

We refine the coordinates of the seed stations according to an objective function that considers both the traffic volume properties of the /24 clusters and the distance of the /24 clusters from their assigned station. Specifically, we use the following procedure:

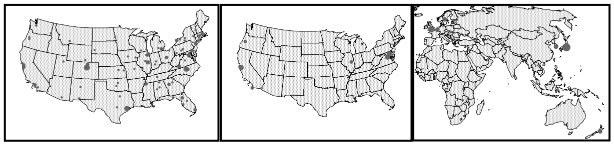
1. Create an initial station location list from the seed CVs.
2. Assign each /24 cluster a weight, which is the number of client requests observed from that /24 cluster.
3. Assign each /24 cluster to the nearest (in terms of network delay) station. Each station and its assigned /24 clusters are a single TC.
4. Calculate the weighted centroid of the CVs of the /24 clusters in each TC.
5. Replace the CV for each station with the weighted centroid for the corresponding TC.
6. If the CVs for multiple stations are within a certain distance (we used 10 ms), select a single CV to represent the group.
7. Repeat steps 3-6 until the station locations remain unchanged across two consecutive iterations or a maximum number of iterations is completed.

The number of clusters resulting from this procedure is generally less than the number of seed stations, since stations that move to within a specified distance of another are merged (in step 6 above).

In the next section, we describe the empirical data used to identify and evaluate traffic clusters. In Section 5, we present the results of our evaluation.

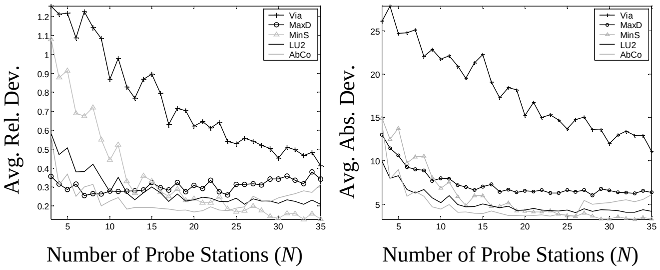
4. EXPERIMENTAL METHODOLOGY

We wish to identify and analyze Web traffic clusters in terms of two key metrics: network proximity and traffic intensity. There were three primary components to our data collection procedure. First, we needed to select the best method to infer distances between arbitrary Internet hosts. We collected data from a set of 113 hosts performing full mesh probing to enable a comparison of actual to inferred distances. This dataset is described in



a) AMP (*I1*) sites b) Skitter ($\mathcal{D}1$) collection sites

Figure 2: Collection sites for *I1*, $\mathcal{D}1$ data. a) The *I1* data set was collected under the Active Measurement Project (AMP) [3]. Of AMP's 130 sites, 113 were used in this study. AMP also has a probe station in Ottawa, Ontario, which was used in this study, but is not shown here. b) The $\mathcal{D}1$ data was collected under the Skitter project [7] from 21 sites. In some cases, multiple sites were too close to plot separately. These sites are depicted as a single, larger marker.



a) Average relative deviation b) Average absolute deviation

Figure 3: Comparison of distance functions for inferring Internet distances. For each value of *N*, the random selection of probe stations was repeated for ten trials. The plotted value represents the average (over ten trials) relative (a) and absolute (b) deviation of samples for a given value of *N*.

Section 4.1. Next, we needed to identify clusters of network proximal Internet addresses. This required measurement data to a very large set of Internet addresses; this dataset is detailed in Section 4.2. Finally, we needed to link these IP addresses to application level traffic generation. The Web traffic data collected for this purpose are presented in Section 4.3.

4.1 Selecting a Distance Inference Technique

The primary goal of our effort was not to propose a new Internet distance estimation technique. However, comparative evaluations of competing proposals [8], [9][10][15] were incomplete due to issues such as small numbers of measurement samples or probe stations. Because an accurate inference mechanism plays a central role in our analysis, we performed an independent comparative evaluation. We compared only the distance functions; we did not evaluate proposed collection infrastructures. We considered only the accuracy of inferring the minimum expected delay between arbitrary end hosts. We provide a summary of this analysis here; the full details of our evaluation are provided in [4].

Our inference evaluation is based on data collected from 113 widely distributed hosts (Figure 2a) performing full-mesh probing under the Active Measurement Project (AMP) [3]. We refer to this dataset as *I1*.

We use minimum RTT, as opposed to average or median RTT, to avoid artificially inflating distance estimates due to congestion. It is also the distance metric used by [8][15]. For *I1* data set, we were consistently able to determine the minimum RTT between all 113 hosts for the entire 30-day period using only 10 randomly selected measurement samples. We recorded the minimum measured delay between all pairings of the 113 hosts. We refer to this measured minimum distance between any two hosts, $\mathcal{H}_1$, $\mathcal{H}_2$ as d($\mathcal{H}_{1,}$ $\mathcal{H}_2$).

We evaluated the five representative distance functions (Via, MaxD, MinS, LU2 and AbCo) described in Section 2.1. We evaluated the performance of each distance function for a given number of probe stations, *N*. Figure 3 provides a plot of the average relative and absolute deviations for each value of *N* tested. All of the distance functions performed fairly well, with average absolute deviations of 25 ms or less for ten or more probe stations. However, calculating distances using absolute coordinates and the Euclidean distance function (AbCo) consistently outperformed the other distance functions in terms of both relative and absolute deviation for 7-23 probe stations.

Based on these results, we use the AbCo method to infer minimum RTT between probed IP addresses for the remainder of this paper.

4.2 Distance Estimation for Arbitrary Hosts

Since the *I1* data set provides distance information only for the 113 AMP stations, we required additional data to estimate distances between arbitrary Internet hosts. The $\mathcal{D}1$ data set was collected for this purpose. The $\mathcal{D}1$ dataset was collected from 21 monitors managed by CAIDA and deployed in spatially and topologically diverse locations [7]. Figure 2b depicts the locations of the 21 monitors. Each monitor is running a non-stop process that probes IP addresses covering significant portions of the routable IP Version 4 address space.

For each of the 21 probe sites, we collected a list of the IP addresses probed, and the minimum RTT observed for that IP address. We considered only addresses for which at least ten RTT measurement samples had been collected, and grouped addresses according to the 24-bit network address prefix. We track only the minimum distance to a /24 cluster, as opposed to individual addresses within the /24 cluster.

Arguably, tracking only the minimum RTT to a /24 cluster, as opposed to individual addresses, could introduce errors into our system. We performed the following informal test before deciding to limit tracking to /24 clusters.

We calculated the coefficient of variation (CoV) of the RTT from a given monitor to all IP addresses within each /24 cluster. We calculated the CoV for all /24 clusters for which we had at least ten measurements to at least five different addresses in the same /24 cluster. The average

CoV was 10.0 – indicating only a small level of variance within a /24 cluster.

There were about 130,000 /24 clusters for which we had ten RTT samples from all 21 monitors. We noted that some of the probed addresses had very large delays (e.g., greater than 300 ms) from all monitors. Minimum delays of greater than 300 ms across all monitors are likely to be associated with, for example, satellite connections or dial-up connections. Since our AMP data did not include such connections, we could not verify the accuracy of known distance inference techniques for such measurements. To avoid skewing our clustering results, we restricted the *D1* dataset to IP addresses with delays of less than 300 ms from at least 1 monitor.

Finally, we selected the set of monitors providing the largest pool of /24 clusters that met the criteria above. We found that with a subset of 11 monitors, we had a usable pool of 71,000 /24 clusters in *D1*.

4.3 Web Traffic Data Sources

We collected traffic data from the Web cache logs of the official websites of the 2002 Wimbledon [2] and US Open [1] events. We refer to these datasets as *W1* and *W2*, respectively. These datasets are representative of the large-scale distributed data delivery services motivating our study.

5. RESULTS

In this section, we present our results in identifying and analyzing Web traffic clusters. Our evaluation focuses on identifying groups of IP addresses that can be treated as a unit, with respect to application level traffic generation and network location.

We begin by using the *D1* and *W1* datasets to compare several seed selection heuristics, and show that the resulting clusters exhibit similar proximity and traffic properties. This implies that the seeding has limited impact on the final cluster properties. We contrast our iterative partitioning technique with schemes focusing on administrative domains, and illustrate significant improvements in the ability to predict network location and traffic patterns. Finally, we demonstrate how our traffic clustering technique can be exploited for intelligent server selection. We use a range of real-world conditions to illustrate scenarios under which clustering provides clear and substantial performance advantages. We also point out conditions under which performance improvements are limited.

5.1 Comparison of Seed Heuristics

Our goal is to identify a representative set of coordinates for the most significant regions of the Internet, in terms of traffic generation. Our technique uses traffic volume heuristics to identify an initial, or seed, list of coordinates, and then refines this list through iterative, partition-based clustering. We begin by comparing the clusters formed when using traffic volume heuristics, to clusters formed using geographic heuristics and clusters formed when seeds are randomly selected.

5.1.1 Seed Selection Heuristics

While there is no authoritative database mapping IP addresses to geographic locations, recent proposals [12][16] describe heuristics to infer the geographic location of IP addresses. We created geographic-based seed sets using one such system. We mapped each /24 cluster in *D1* to a geographic location via IxMapper/GLS [12]. We compiled a list of all the locations at which we found at least two /24 clusters within 5 ms of each other. We calculated the centroid of the /24 clusters for each location that met this criteria. If a cluster was within 5 ms of other clusters, we selected a single representative location, by calculating the centroid of the locations. We found 157 such locations. We repeated this test by searching for locations with two or more /24 clusters within 10 ms of each other. We found 2615 such locations. We refer to the former set of CVs as "small#geo," and the latter set as "large#geo."

We also created two seed sets using the traffic volume heuristics described in Section 3.1. Here, the CVs with the largest traffic volumes were selected to serve as seeds. We refer to the set with the top ranked 157 CVs as small#vol, and the set with the top ranked 2615 CVs as large#vol.

Finally, we randomly selected a set of 157 and 2615 coordinates from *D1* to form the small#ran and large#ran seed sets. We compared the clusters identified using the random and geographic seed sets to those formed using the traffic volume heuristics.

5.1.2 Evaluation of Seed Selection Alternatives

In Figure 4, we plotted the distribution of traffic generated by /24 clusters, with respect to the distance from their assigned station. For example, using clusters formed from the large#ran seed set, over 70% of the client requests were generated from locations within 10 ms of their assigned station, and over 96% from within 20 ms.

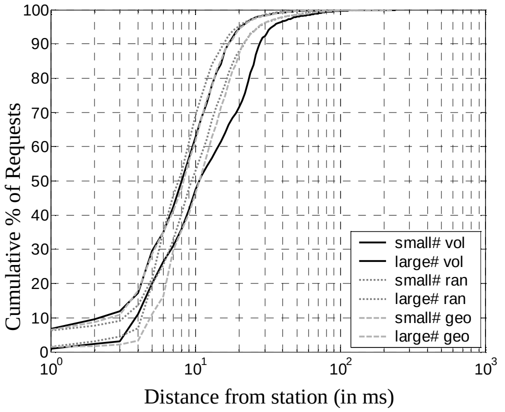
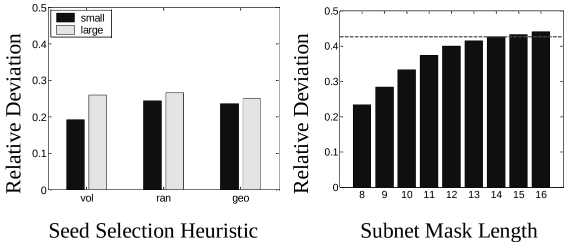


Figure 4: Cumulative distribution of client requests (traffic volume) to network distance. The x-axis is in log scale for improved readability. Small# clusterings used 157 seeds; large# clusterings used 2615 seeds. Plots for large#vol and large#geo are overlapping in many areas.



(a) (b)

Figure 5: Maximum relative deviation between actual and predicted number of client requests using standard linear prediction. a) The bars compare traffic volume, random and geographic seed selection heuristics. Small clusterings used 157 seeds; large used 2615 seeds. b) Comparison of partitions formed using indicated subnet mask length. The horizontal dashed line represents the deviation of the BGP partitioning.

Clusters resulting from the large#vol and large#geo seed sets had similar proximity results. Expectedly, fewer clusters (as formed from small#geo, small#ran, small#vol), resulted in wider clusters. For example, approximately 87% of the client requests under small#geo and small#ran were generated from locations within 20 ms of their assigned station.

We also evaluated our ability to predict the traffic of these six partitionings, by applying standard linear prediction to the number of requests generated by each cluster. The linear prediction used a sliding window of ten 60-second intervals to predict the expected number of requests for the next 60-second interval. We calculated the relative deviation between the actual and predicted number of requests for each interval in *W1*.

Figure 5a compares the maximum relative deviation incurred by TCs derived from each seed set. (Figure 5b is discussed in Section 5.2.) Partitionings with smaller numbers of TCs had consistently lower relative deviations than partitionings with larger numbers of TCs. We also tested other prediction schemes (such as simple moving average, exponentially weighted moving averages and best linear fit) and prediction parameters (such as longer and shorter windows and prediction intervals), but found that simple linear prediction, with three coefficients, produced the lowest relative deviation values in this environment.

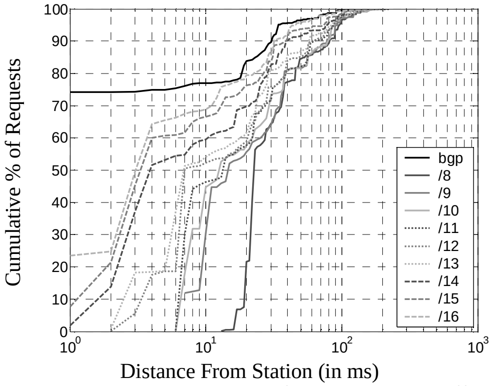


Figure 6: Cumulative distribution of client requests (traffic volume) to BGP and subnet clusters. The x-axis is in log scale for improved readability.

Together, Figures 4 and 5a show that using a large number of TCs can improve accuracy in estimating client locations, but it also reduces the TC client population, which may make traffic prediction more difficult. In later sections, we make this trade-off more concrete by comparing clustering alternatives when used for intelligent server selection. However, we first compare the proximity and traffic prediction of our TCs to clusters formed using BGP routing data and IP address prefixes.

5.2 Topological clustering comparison

We compared our TCs to clusters formed using only network topological data. We refer to partitionings formed from topological data as BGP clusters and subnet clusters.

For the BGP cluster partitioning, we collected a list of IP address prefixes from the BGP routing tables of [19] for the time period corresponding to the *W1* event. We then assigned the IP addresses in *D1* to the corresponding BGP prefix. We used the centroid of the addresses assigned to each BGP cluster as the coordinates for each BGP cluster.

We created several subnet partitionings to reflect a range of fixed subnet mask lengths. That is, we began by assigning all IP addresses in *D1* with the same first 16-bits to the same cluster. We refer to this partitioning as the /16 partitioning. We used the centroid of the addresses assigned to each /16 cluster as the coordinates for the cluster. We repeated this process to form 9 partitionings, with mask lengths ranging from 8 to 16 bits.

Figure 6 compares the network proximity properties of BGP and subnet clusters. The plot for BGP clusters is misleading however. It indicates approximately 73% of the requests were generated within 0 ms (i.e., from the exact location) of the assigned station location. On closer inspection, we noted that 73% of the clusters had only a single /24 address within the cluster. Thus, the apparent tight proximity of the BGP clusters is an artifact our limiting probe data tracking granularity to /24 clusters, as opposed to a benefit of BGP clusters.

Moreover, the large number of small BGP clusters made traffic prediction less accurate. Figure 5b compares the maximum relative deviations between actual and predicted number of requests for BGP and subnet clusters. BGP partitioning has higher relative deviations than all but the /15 and /16 subnet partitionings.

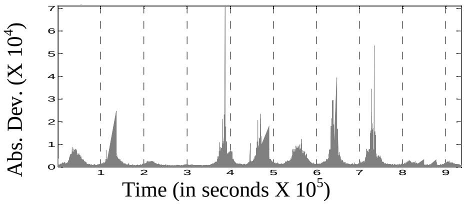
We had anticipated better traffic prediction capabilities for subnet partitionings with smaller numbers of clusters (e.g., /8 and /9), than for partitionings with larger number of clusters (e.g., BGP and /16). However, we were somewhat surprised that the average relative deviation for the /8 partitioning was comparable to that achieved under the traffic clustering schemes. We decided to investigate this more thoroughly by tracking the relative deviation over the course of the *W1* event.

We compared the absolute deviations of the traffic predictions achieved under the /8 subnet clustering to those achieved under traffic volume clustering. We chose to compare TCs formed using the traffic volume seed selection heuristic for 500 seed stations, as its average deviation was closest to that of the /8 partitioning. The /8 partitioning had 87 clusters and an average relative deviation of 0.24; the traffic volume partitioning had 145 clusters and an average relative deviation of 0.25. We compared total absolute deviation, as opposed to maximum relative deviation, to gain better insights into the deviation across all clusters. As Figure 7 shows, our TCs result in substantially more predictable traffic patterns – the total absolute deviations for /8 subnet clusters are roughly 2 orders of magnitude larger than those of our TCs.

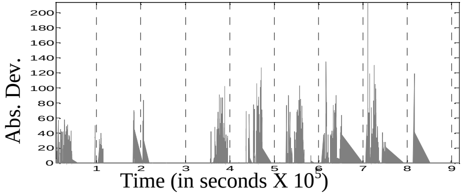
5.3 *W2* cluster properties

We were also interested in whether TC properties hold over time, across events. To investigate, we considered a second event, *W2*. The traffic volume for *W2* is substantially larger than for *W1*. Also, while both events are considered of international interest, the events were staged on different continents, about 2 months apart.

We ran the same procedures described in Section 3 to identify the initial set of stations based on traffic volume, and iteratively refine this station list. However, for this test, we used the *W2* dataset. We again selected 157 and 2615 seeds. We refer to the resulting partitionings as small#w2 and large#w2. We found that traffic prediction, using the same linear prediction method described for *W1*, performed almost identically in terms of relative deviation.



a) Total Absolute Deviation for /8 clusters



b) Total Absolute Deviation for medium# vol clusters

Figure 7: Comparison of the absolute deviation between the predicted and actual request volume under subnet and vol clusterings. a) is the sum of the absolute deviations across all of the clusters obtained under the /8 subnet method. b) is the sum of the absolute deviations across all clusters obtained with 500 (medium#vol) seed stations. Note the maximum y-value plotted in a) is 7.1×10^4; the maximum y-value in b) is 214. The peaks in each graph correspond to surges in the overall request rate.

However, due to the larger client population, a larger number of clusters were identified for each partitioning. Small#w2, which started with 157 seeds, resulted in 111 clusters (compared to 47 clusters identified for *W1*). The small#w2 clients were not as tightly clustered around the station locations as the small#vol TCs for *W1* had been. For example, only 88% of the client requests were generated from locations within 30 ms of their assigned stations. For small#vol, 95% were within 30 ms. The results for large#w2 and large#vol were comparable – 95% of the client requests were generated from within 20 ms of their assigned stations for both partitionings.

5.4 Application: Intelligent Server Selection

Currently, intelligent server selection and network control mechanisms make only limited use of the traffic generation properties of Internet clusters. If given the ability to predict the load imposed by Internet clusters, these algorithms could better map workloads to server and network resources. In this section, we provide a simplified, but nevertheless illustrative, example of the impact traffic clustering can have on intelligent server selection.

Commercial server set deployments often use a greedy strategy [5], in which clients are directed to the server expected to result in the lowest network delay. We used this greedy strategy and the IP addresses of commercially deployed Web servers discovered in [5] to create a simulation environment. We implemented the greedy selection strategy such that when a client request is received, the client's cluster is identified and the server with the lowest expected network delay is selected from those servers that were not overloaded. Each assignment was then evaluated. If the assignment resulted in server overload or the expected delay between the client and the assigned server exceeded a specified threshold, we considered the request "misdirected." Note that assignment is performed based on the cluster CV, whereas evaluation is performed using the CV of the individual client. Clustering that more accurately reflects client location is less likely to assign clients to a server exceeding the delay threshold. Additionally, clustering that more accurately predicts cluster traffic volumes is less likely to overload servers.

We measured the results when the server selection algorithm used the large#vol TCs to direct client requests from *W1*. We compared the results to those achieved when the server selection algorithm used the BGP clusters and /8 subnet clusters identified in Section 5.2. Figure 8 depicts this comparison. For each of the clustering techniques, the number of misdirected clients decreases as more servers are deployed. This is expected since an

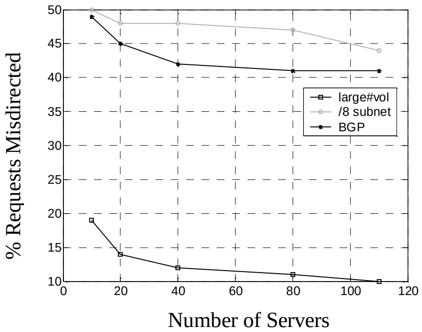


Figure 8: Comparison of server selection results for traffic, BGP and /8 subnet clustering. Client requests are considered "misdirected" if the request is directed to a server that is overloaded or at a network delay greater than the requested threshold. For this test, the delay threshold was set to 50 ms.

increase in the number of server locations results in an increase in the number of servers within the delay threshold of each client.

The most striking feature of Figure 8 however, is that using our TCs, the server selection algorithm misdirects less than half as many requests as with the BGP and /8 subnet clusterings. For example, compare the results when ten servers were used. BGP and /8 subnet clusterings resulted in 48% and 50% of the clients (respectively) being directed to a server that was either overloaded, or located more than 50ms away. Our large#vol clustering reduced misdirection to 19% of the client requests.

6. CONCLUSIONS

In this paper, we developed a methodology to identify and characterize Web traffic clusters. Our approach combines techniques from network-aware clustering, location inference, spatial analysis and traffic modeling. We depicted the properties of Internet clusters through a large, measurement-based evaluation that shows we can achieve good prediction capabilities and network proximity for the large-scale Web events motivating this study. Moreover, we demonstrated that our clustering technique results in a substantial improvement in the ability of a typical server selection algorithm to effectively direct client requests to service points that meet network delay and server load criteria.

While our study is performed in the context of Web traffic, it is not limited to a single application model. We expect our clustering techniques will be useful in addressing a variety of network and distributed server management issues, and for effective new simulation techniques.

7. ACKNOWLEDGMENTS

Anees Shaikh and kc claffy made many valuable suggestions. Daniel Anderson, Andre Broido, and Young Hyun provided access and info for the Skitter (*D1*) dataset. We are thankful for all of their help.

8. REFERENCES

[1] 2002 US Open Tournament Web Site, August 28 – September 5, 2002, http://www.usopen.org.

[2] 2002 Wimbledon Tournament Web Site, June 27 – July 7, 2002, http://www.wimbledon.org.

[3] Active Measurement Project. http://amp.nlanr.net. Funding for this work was provided by the National Science Foundation Cooperative Agreement No. ANI-9807479, and the National Laboratory for Applied Network Research

[4] L. Amini, "Models and Algorithms for Resource Management in Distributed Computing Cooperatives," PhD thesis, Columbia University, 2003.

[5] L. Amini, A. Shaikh, H. Schulzrinne, "Modeling Redirection in Geographically Diverse Server Sets," *World Wide Web Conference*, May 2003.

[6] S. Bhattacharjee, Z. Fei, "A Novel Server Selection Technique for Improving the Response Time of a Replicated Service," *IEEE Infocom*, March/April 1998.

[7] CAIDA's Internet Topology Data Kit #0204. http://www.caida.org. Funding for this work was provided by DARPA's Network Modeling and Simulation Program N66001-01-1-8909.

[8] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, L. Zhang, "IDMaps: A Global Internet Host Distance Estimation Service," *IEEE/ACM Transactions on Networking*, October 2001.

[9] K. Gummadi, S. Saroiu, S. Gribble, "King: Estimating Latency between Arbitrary Internet End Hosts," *Internet Measurement Workshop* November 2002.

[10] S. Hotz, "Routing Information organization to support scalable interdomain routing with heterogeneous path requirements," PhD Thesis, University of Southern California, 1994.

[11] B. Huffaker, M. Fomenkov, D. Plummer, D. Moore, k claffy, "Distance Metrics in the Internet," *IEEE International Telecommunications Symposium*, September 2002.

[12] IxMapper. http://www.ixiacom.com/products.

[13] A. Jain, R. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1995.

[14] B. Krishnamurthy, J. Wang, "On network-aware clustering of web clients," *ACM SIGCOMM*, August 2000.

[15] T. Ng, H. Zhang, "Predicting Network Distance with Coordinates-Based Approaches," *IEEE Infocom*, June 2002.

[16] V. Padmanabhan, L. Subramanian. "An Investigation of Geographic Mapping Techniques for Internet Hosts," *ACM SIGCOMM*, August 2001.

[17] S. Ratnasamy, M. Handley, R. Karp, S. Shenker, "Topologically-Aware Overlay Construction and Server Selection," *IEEE Infocom*, June 2002.

[18] "Route Control: Building Real Confidence in the Internet," http://www.telechoice.com, February 2002.

[19] RouteViews. http://www.antc.uoregon.edu/route-view.