Clustering Hosts in P2P and Global Computing Platforms

Abhishek Agrawal [1] Henri Casanova [1,2]

[1] Dept. of Computer Science and Engineering
[2] San Diego Supercomputer Center
University of California, San Diego

Abstract—

Being able to identify *clusters* of nearby hosts among internet clients provides very useful information for a number of internet and p2p applications. Examples of such applications include web applications, request routing in peer-to-peer overlay network, and distributed computing applications. In this paper, we present and formulate the internet host clustering problem. Leveraging previous work on internet host distance measurement, we propose two hierarchical clustering techniques to solve this problem. The first technique is a *marker* based hierarchical partitioning approach. The second technique is based on the well known K-means clustering algorithm. We evaluated these two approaches in simulation using a representative Internet topology generated with the GT-ITM generator for over 1,000 hosts. Our simulation results demonstrate that our algorithmic clustering approaches effectively identify clusters with arbitrary diameters. Our conclusion is that by leveraging previous work on internet host distance estimation, it is possible to cluster internet hosts to benefit various applications with various requirements.

I. Introduction

As the internet infrastructure becomes more pervasive and connects increasingly powerful commodity resources, many diverse and complex applications attempt to harvest the potential of that infrastructure. For many of these applications, being able to identify hosts that are relatively close to each other in the internet, i.e. identify host *clusters*, can provide ways to improve both performance and scalability. For example, a web service provider would like to place object replicas close to such internet host clusters and hence reduce the service latency perceived by clients as well as the load on the web server itself [2]. Peer-to-peer routing protocols such as Chord [15], Pastry [14], and Tapestry [17] need to maintain sets of nearby peers in order to route queries more efficiently. Another application domain for which clustering is becoming an important issue is that of distributed computing. Although most currently deployed applications (e.g. SETI@home [3]) utilize a centralized server to run embarrassingly parallel applications, identifying host clusters makes it possible to address issues of data locality for data-intensive applications. Also, host clustering makes it possible to execute part of a parallel application within a cluster. Indeed, low intra-cluster network latencies can enable coarse-grain parallelism which is not feasible among an arbitrary set of internet hosts.

Internet host clustering strategies that have been proposed in the past, are, in general, application specific. Instead of attempting to solve the problem for a particular application, we generalize the problem to finding optimal clusters in the internet graph. The works in [6] and [12] propose architectures that enable the generation of an approximate internet *distance map*. That is, it is possible to estimate the approximate distance between any two hosts on the internet. The distance metric corresponds to the latency in the IP network. This enables the modeling of the internet hosts as nodes of a weighted graph, where the weights correspond to the relative host-to-host distances in the distance map. The host clustering problem can then be defined as finding an optimal partitioning for this graph.

As such, the equivalent graph-theoretic formulation of host clustering is a *hard* problem. The problem can be formulated as follows: Given a weighted Graph $G(V,E)$ and a distance D, partition the graph into K clusters such that K is minimal, the clusters cover the entire graph and the distance between any two nodes in any cluster is less than D. The equivalent decision problem (Does graph $G(V,E)$ have $k < K$ partitions of diameter D?) is known to be NP-Complete [7].

In this paper, we propose two heuristics to find an approximate solution to the internet host clustering problem and its equivalent graph theoretic formulation. We claim that, given the nature of most internet applications, which can potentially benefit from a host clustering strat-

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

egy, a non-optimal heuristic solution is acceptable. Our two heuristics leverage the work in [6] and [12] respectively, as internet host distances metrics are the basis for host clustering. Our simulation results demonstrate that our approach leads to effective internet host clustering, with cluster diameters being below the diameter required by an application with high probability.

The rest of this paper is organized as follows. In Section II, we discuss related work. We present our host clustering heuristics in Section III. Our evaluation strategy and simulation results is presented in Section IV & V. Finally, we discuss some open issues in section VI and conclude in section VII.

II. Related Work

The simplest proposed approach to clustering web clients is based on prefix length matching of IP addresses (e.g. obtained from access logs of web servers). Clients which share the same n bits of IP addresses are said to belong to the same network cluster. The quality of clustering obtained by this approach is questionable because network address prefixes may have varying lengths and clients with same prefix may end up in different geographic networks. A modified approach proposed in [10] uses network prefixes along with BGP network mask information, which provides a better clustering. The work in [5, 8] proposes the use of internet Domain Name Service to cluster the clients. Clients which use the same local DNS server for address resolution are clustered together. However, the above methods rely on specialized information like web server traces or DNS usage pattern to determine the clustering. Such information is not directly available to end-user applications. Also, such strategies lead to more static clusters which do not vary much over time.

Our approach aims to provide generic clustering strategies, which can be tuned using input parameters to generate application-specific clusters. Our hierarchical strategy bears similarities to the approach presented in [13]. They propose a binning strategy, in which all hosts, measure their distance from a set of well-known landmark nodes. Each node orders the landmarks based on the distance from them. All nodes with same order are binned together. Our approach clusters the hosts based on a user tunable diameter D, and generates clusters in a hierarchical fashion. Another difference between our work and that in [13] is that our approach aims to leverage existing distance estimation infrastructure, and can be adapted to complement the approach presented in [13].

III. Algorithmic Approach

In this section we present our algorithmic approach to tackle the host clustering problem. We first discuss previous approaches to build approximate *internet distance maps*, i.e. to provide a way to estimate the IP latency among any two internet hosts. We then present two clustering strategies, which take an internet distance map as input and heuristically partition the network into clusters.

A. Distance Maps and Clustering

A well-known solution to the internet distance prediction problem was provided by Francis *et.al*. [6], through the IDMaps service. IDMaps is an infrastructure service in which special HOPS servers maintain a virtual topology map of the internet consisting of end hosts and some special hosts called Tracers. The distance between hosts A and B is estimated as a distance between A and its nearest Tracer T_1, plus the distance between B and its nearest tracer T_2, plus the shortest distance from T_1 to T_2 over the tracer topology. IDMaps has been designed as a client server architecture and end hosts can query the HOPS servers to obtain network distance predictions and build a approximate host distance map.

The work in [12] proposes an alternative architecture for network distance prediction called Global Network Positioning(GNP). They propose a *coordinates-based* approach to network distance prediction in the peer-to-peer architecture. Each host in the network is assigned a set of coordinates in a multi-dimensional coordinate space. Thus, the network distance between two hosts can be predicted by evaluating a distance function, such as Euclidean distance, over the hosts coordinates. As a result, the coordinate based approach directly provides a distance map for internet hosts.

Given a distance map, the **host clustering problem** can be formulated as: Given a connected weighted Graph $G(V,E)$ where the vertices represent hosts and the weighted edges represent distances according to the distance map, and a fixed diameter D, partition the graph into K clusters such that K is minimal, the clusters cover the entire graph, and the distance between any two nodes in any cluster is less than D. We call K the optimal number of clusters.

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

Algorithm : FIND MARKERS()

Choose the first marker randomly
repeat
Find a host at least
a distance D from all current markers
if *one such host is found*
Add it to the marker set
until *marker set changes*

Fig. 1. Algorithm for determining marker hosts.

B. Clustering Strategies

We propose two algorithms to identify clusters in a given topology. Both algorithms make a simplifying assumption that diameter constraints are not hard. More specifically, we tolerate clusters in which hosts are within a distance D of each other with high probability. We believe that this assumption is reasonable for most internet applications.

Both algorithms use a set of distinguished hosts called *markers*, which need to be at least a distance D from each other. The objective is to find as many markers are possible, the rationale being that the number of such markers is a lower bound on the optimal number of clusters. Finding a maximum number of makers is in itself NP-hard. The next section describes a simple heuristic for choosing markers.

B.1 Marker Hosts

We apply the strategy depicted in Figure 1 for selecting markers in a topology. The strategy chooses as many markers as possible, which are at least a distance D apart from each other. Thus, the number of markers is determined dynamically and therefore becomes a function of the input topology and diameter D. The strategy is justified since the number of markers chosen by the above process is actually a lower bound on the optimal number of clusters of diameter D. In the worst case, marker determination takes $O(N*M)$ host-to-host distance estimations, where M is the final number of markers. Note that M is bounded by the number of clusters, which is orders of magnitude lower than the total number of hosts for reasonable values of D. In the scope of this paper, the host-to-host distance estimation comes directly from the input distance map. Note that this marker selection algorithm could be implemented in practice on the internet with the same complexity (e.g. via ping measurements).

B.2 Hierarchical Clustering

Given a generic distance map, we propose the following *hierarchical clustering* algorithm. The algorithm proceeds in three stages. In the first stage, a set of markers are selected according to the procedure described in the previous section. In the second stage each host obtains its internet distance from all the markers (according to the distance map). For each host one can then determine the marker nearest to that host. All hosts nearest to the same marker are clustered together. In the third stage the average diameter of each cluster is estimated again according to the distance map. If the diameter of a cluster does not lie within the acceptable range D with high probability, then the algorithm gets invoked again to sub-partition this cluster into smaller sub-clusters.

B.3 K-means Clustering

As described above, The GNP approach [12] assigns coordinates to each host in a multi-dimensional coordinate space and Euclidean distance can then be used to estimates distances in the IP network. This makes it possible to treat host clustering as a generic data clustering problem such as the ones encountered in other domains (e.g. databases, AI). A popular heuristic for discovering clusters is the K-means algorithm. Given GNP coordinates, we run K-means in a hierarchical fashion to partition the multi-dimensional space. The algorithm proceeds in three stages. First a *near optimal* number of initial cluster-heads is chosen. This is done by choosing a set of markers as explained in Section III-B.1. These chosen cluster-heads are given as input to the K-means clustering algorithm. Each point is initially added to a cluster centered around the nearest cluster-head. Next, the centroid of each cluster is calculated and the points are reclustered based on the distance from the nearest centroid. This process is repeated till the clusters converge. In the third stage, the average diameter of each cluster is computed. If the diameter of a cluster does not lie within the acceptable range D with high probability, then the algorithm is reinvoked to obtain sub-clusters.

IV. EVALUATION STRATEGY

In this section we present the evaluation strategy, our simulation model. In the next section, we present simulation results that show the feasibility of our algorithmic approaches.

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

In order to evaluate our strategies we use the following two metrics.

• *Average Advantage Ratio:* We first compute the average inter-cluster latency, i.e average communication latency between two nodes belonging to different clusters. We also compute the average intra-cluster latency, i.e average communication latency between two nodes between the two nodes in the same cluster. The average is taken over all such node pairs. The Advantage-ratio is then the ratio of the inter-cluster latency to the intra-cluster latency.

• *Average Coefficient of Variation:* For each cluster and for each node outside that cluster, we measure the mean and standard deviation of the latency from nodes within the cluster to the node outside the cluster. The ratio of the standard deviation to the mean is defined as the coefficient of variation. The average is computed over all clusters.

Let us now explain what these metrics convey. The first metric is similar to the one used in [13]. Intuitively, it is a measure of the advantage achieved, in terms of communication latency, when communicating with a node within one's cluster instead of a node outside the cluster. The advantage ratio is also a description of the network view as seen by nodes within a cluster. I.e, it describes how much closer a node is to its neighbors in its cluster compared to the rest of the network. The rationale behind the second metric is as follows. Nodes that lie within the same cluster should have similar latency values to a node outside the cluster. Hence, the lower the coefficient of variation in these latency values, the better the quality of the cluster is. The second metric, in some sense, describes how the clusters are viewed by the rest of the network. The ideal metric to evaluate any heuristic clustering strategy is the number of valid clusters returned by the algorithm. The smaller this number, the closer the heuristic is to the optimal. However, when we consider the internet host clustering problem, and the set of potential applications that can leverage a solution to it, the ideal metric, in itself, is not very useful. We therefore, evaluate our approaches using only the above two metrics. However, it remains an interesting exercise to evaluate the heuristics in terms of the ideal metric.

We implemented both our algorithms and used simulation to evaluate them. The GT-ITM [16]topology generator was used to generate a transit-stub topology with over 1000 nodes, with total diameter of around 300 ms. The topology had six transit domains with eight transit nodes per domain. Each transit node had three stub domains attached to it with each stub containing eight stub nodes on an average. All our simulation results were obtained on this topology. The distance map corresponding to IDmaps was generated by placing tracers on transit nodes of the GT-ITM graph. We ran the GNP simulation tool, with the GT-ITM topology as input, to obtain a coordinate distance map. The parameters to the GNP tool where chosen to generate coordinates in a 2-dimensional Euclidean space. It is important to note here that our clustering algorithms do not use the perfectly accurate distances that can be obtained directly from the topology we simulate, but rather use measurements obtained with either IDmaps or GNP on that topology.

For both our clustering strategies, a valid cluster was returned if it had a diameter D with probability $p > 0.8$. We ran our clustering algorithm on the generated distance maps. The diameter was varied from 40 ms to 200 ms and each time the advantage ratio and the coefficient of variation was computed.

V. Simulation Results

The results of the simulations are presented in Figures 2 and 3. Figure 2 plots the advantage ratio for different diameter values. For both heuristics, we observe that the advantage ratio is high for low values of diameter and drops off as the expected diameter is increased. This is expected, since, as we increase the required diameter, the intra-cluster latency increases, thereby decreasing the advantage ratio. We also plot the expected advantage ratio for various diameter. This is obtained by dividing the average inter-cluster latency by the value D. We observe that the actual advantage is higher than the expected one. This indicates that the actual intra-cluster latencies are lower than D. The advantage ratio, similarly to the metric used in [13], simply shows that the obtained clusters satisfy the diameter constraint.

Figure 3 plots the average coefficient of variation, measured as described in the previous section, for different values of the required diameter. We believe that the coefficient of variation is a better way to evaluate clustering than the advantage ratio. Indeed, a low value of this metric implies that each cluster is seen as such by the rest of network. The results show that, for both heuristics, the variation in the latencies to a random node outside the cluster is about 3-10 %, and increases slightly as the diameter is increased. For a large topology, such as the actual internet, a 3-10% variation is comparable to

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

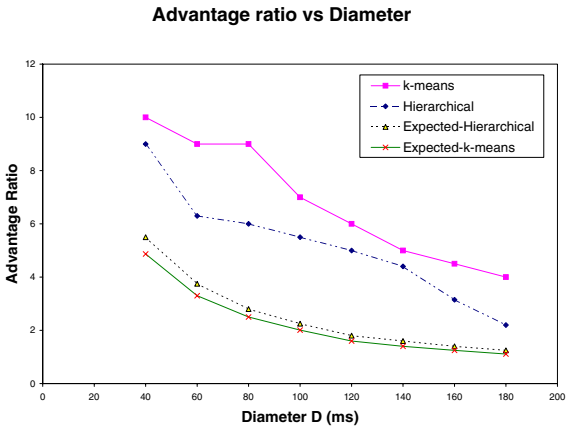

Fig. 2. Avg. Advantage Ratio vs. diameter.

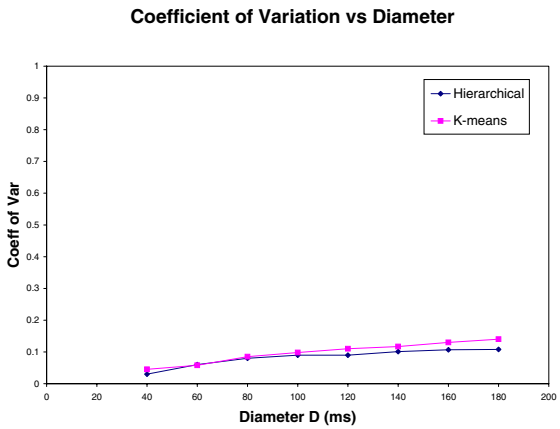

Fig. 3. Avg. Coeff. of Variation vs. diameter.

variations in latency due to transient network behavior.

The above results indicate that our two clustering strategies are effective for clustering internet hosts. An interesting question is that of the impact of distance map accuracy on the effectiveness of clustering. In our experiments we have performed clustering on the distance maps generated by either IDmaps or GNP. These distance maps are only approximations of actual host-to-host latencies. The impact of this approximation is seen for small values of D, for instance due to the triangulation errors incurred by IDmaps. More specifically, if D is comparable to intra-stub latencies or lower, then our clustering strategy will generate too many clusters. As mentioned in Section IV, it would be interesting to quantify the clustering quality by comparing with an optimal solution computed over the perfectly accurate distance map (e.g. directly from the topology). We shall consider this in our future work.

VI. Discussion

In this paper we have presented and evaluated algorithmic strategies which can leverage existing internet distance measurement infrastructures to solve the internet host clustering problem. Our simulation results indicate that even simple strategies such as the ones presented in this paper actually do a reasonably acceptable clustering job. However, there remain a few open issues which merit further discussion.

Both our clustering strategies are different from other internet host clustering techniques, in that they allow the application to specify a diameter D, such that with high probability, all nodes within a cluster are at distance D from each other. We argue that this is a useful feature to have since different applications can have different requirements depending on which the choice of the clusters will depend. For example, a distributed computing application may want to schedule sets of loosely communicating tasks on hosts which are not too geographically distant. The value of D can then be used to determine the right choice for clustering given that communication requirements of the target application. Also, a number of organizations (e.g. CAIDA [4]) measure inter and intra continental internet latencies. Applications can use these to get an estimate of the value of D that should be used for clustering.

Another important aspect of both our strategies is the choice of marker hosts. The quality of the marker set determines the overall quality of the clusters. It will be interesting to experiment with different algorithms for marker host selection and see its impact on clustering. In particular, we expect our two approaches to exhibit different sensitivity levels to the quality of the original marker set. Our intuition is that the K-means strategy is more robust than the hierarchical strategy because K-means iteratively recomputes clusters and can therefore "escape" more easily from bad initial conditions. The algorithm presented in Figure 1 has a O(N*M) complexity where N is the total number of hosts and M is the number of chosen markers. One way to reduce this complexity, when considering actual deployment, is to select the markers from a list of well-known internet end points, which are well spread out in the network. For example, such a list could be made up of university servers, DNS servers, each located at sufficiently large distance from each other.

We have so far considered only the algorithmic as-

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

pects of our approach. The issue of how to actually deploy our strategy remains an open one. Both, centralized and distributed, deployments have their relative pros and cons. The actual choice, we believe, would depend on the set applications which will use such a clustering strategy. Although our clustering strategies assume the existence of a internet distance estimation infrastructure like IDMaps or GNP, the strategies themselves are infrastructure independent and can be used in conjunction with any such estimation infrastructure. It is important to note that, for the Hierarchical clustering strategy, once the marker nodes get chosen, the nodes need to estimate their distance to these marker nodes only. Therefore, a complete distance map information is actually redundant. This strategy is similar to the the one presented in [13] in that it can be directly deployed on the internet, without requiring a distance estimation infrastructure.

Finally, the choice of the actual clustering algorithm one can use is very broad. The relative success of simple strategies such as ours indicates that more sophisticated strategies, such as those from the Artificial Intelligence domain, can be employed to obtain better results.

VII. Conclusion

In this paper, we presented the host clustering problem. Two heuristic strategies, hierarchical clustering and k-means clustering, which leverage internet distance estimation infrastructure, were described. Our simulations with a GT-ITM topology indicate that the presented strategies are effective for clustering internet hosts with various requirements for intra-cluster latencies. It will be interesting to corroborate these results by running similar simulations with more GT-ITM topologies, with ones generated by other generators (e.g. with BRITE [11]), and with actual internet topologies (e.g. from NLANR [1]). In particular, exploring a range of topologies will allow us to compare the merit of our two clustering approaches more conclusively.

The next step in this research is to precisely quantify the benefit of host clustering with our approach for actual internet applications. In particular, we will focus on the use of internet platforms for deploying scientific computing applications. For these applications, identifying host clusters makes it possible to address issues of data locality. Also, host clustering makes it possible to execute part of a parallel application within a cluster. Indeed, low intra-cluster network latencies can enable coarse-grain parallelism which is not feasible among an arbitrary set of internet hosts. In previous work [9] we have studied how systems like SETI@home [3] could be extended to support parallel bio-informatics applications, with a focus on performance. By providing host clustering methodologies, this work provides a basis for further enhancing application performance and overall system throughput. This will however mandate more evolved evaluation models that encompass link bandwidth characteristics, bandwidth sharing properties, and potential asymmetry.

References

[1] Active Measurement Project (AMP). `http://watt.nlanr.net`.

[2] Akamai. `http://www.akamai.com`.

[3] SETI@home. `http://setiathome.ssl.berkeley.edu`.

[4] Cooperative Association for Internet Data Analysis. `http://www.caida.org`, 2001.

[5] Bestavros, A., and Mehrotra, S. DNS-based Internet Client Clustering and Characterization. In *Proceedings of the 4th IEEE Workshop on Workload Characterization (WWC'01), Austin, TX* (December 2001).

[6] Francis, P., Jamin, S., PAxson, V., Zhang, L., Gryniewicz, D., and Jin, Y. An architecture for global internet host distance estimation service. In *Proceedings of IEEE INFOCOM'99, New York, NY* (March 1999).

[7] Garrey, M. R., and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, CA, 1979.

[8] Gummadi, K. P., Saroiu, S., and Gribble, S. D. King: Estimating Latency between Arbitrary Internet End Hosts. In *Proceedings of SIGCOMM IMW 2002, Marsille, France* (November 2002).

[9] Kondo, D., Casanova, H., Wing, E., and Berman, F. Models and Scheduling Mechanisms for Global Computing Applications. In *Proc. of the International Parallel and Distributed Processing Symposium (IPDPS), Fort Lauderdale, Florida* (April 2002).

[10] Krishnamurthy, B., and Wang, J. On Network-Aware Clustering of Web Clients. In *Proceedings of ACM SIGCOM, New York, NY* (August 2000).

[11] Medina, A., Lakhina, A., Matta, I., and Byers, J. BRITE: An Approach to Universal Topology Generation. In *Proceedings of the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01)* (August 2001).

[12] Ng, T. E., and Zhang, H. Predicting Internet Network Distance with Coordinates-based Approaches. In *Proceedings of IEEE INFOCOM'02, New York, NY* (June 2002).

[13] Ratnasamy, S., Handley, M., Karp, R., and Shenker, S. Topologically-Aware Overlay Construction and Server Se-

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

lection. In *Proceedings of IEEE INFOCOM'02, New York, NY* (June 2002).

[14] ROWSTRON, A., AND DRUSCHEL, P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany* (November 2001).

[15] STOICA, I., MORRIS, R., KARGER, D., FRANS KAASHOEK, M., AND BALAKRISHNAN, H. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of ACM SIGCOMM'01, San Diego, California* (August 2001).

[16] ZEGURA, E., CALVERT, K., AND BHATTACHARJEE, S. Hoe to Model an Intertnetwork. In *Proceedings of IEEE INFOCOM'96, CA* (May 1996).

[17] ZHAO, B., KUBIATOWICZ, J., AND JOSEPH, A. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Tech. Rep. UCB/CSD-01-1141, University of California, Berkeley, 2001.

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE