Multi recipient anycast routing

Lawrence Latif, Eleni Mykoniati, Raul Landa, Richard Clegg, David Griffin, Miguel Rio

† University College London

Abstract: This paper describes the design of the necessary mechanisms of a new network primitive: n-casting, the transmission of a message to the closest *n* members of a given group. This primitive can be use for anycasting (n=1), p2p swarming, replicated databases querying and multicast among other applications. We describe a new hierarchical clustering method to allow nodes to have an image of the Internet which is accurate and requires little storage, a new data structure which compresses group membership with decreasing accuracy as *n* grows and a new protocol to exchange information. These mechanisms can allow implementation of n-casting at the application or network layer.

1 Introduction.

The anycast protocol [4] allows a packet to be routed to a single member of an anycast group determined by a network or application layer metric. Here we present n-casting, an application layer anycasting protocol that provides the ability to message n members ($n \geq 1$) of an anycast group in order of geographical distance, allowing for a quality-of-service (QoS) orientated overlay.

Anycast routing can be implemented at either network or application layers and has seen widespread deployment in the domain name system (DNS) [3]. Outside of DNS, Castro et al. [5] state that anycast is a "powerful building block" for peer-to-peer (P2P) networks. The ability to discover multiple group members allows overlay nodes to target nodes that are best placed, geographically, to answer queries.

At the network layer, anycast's inability to aggregate group addresses means there is significant resource demands on routers, a demand that can be mitigated by implementing anycast as an application layer protocol. Though work has been done [2] to improve the scalability of network layer anycast, at present application layer anycast appears more promising for applications such as P2P swarming, database querying and multicast. At the application layer constraints on resources such as memory is significantly lower, allowing for overlays with a large number of groups and high membership.

Application layer anycast introduces the possibility of using non-network level metrics in order to decide the destination of the anycast packet. The need to consider other application level metrics was highlighted by Zegura et al. [6], when they proposed an anycast DNS service where nodes would query a metric database to provide a suitable node, much like unicast DNS. In experiments Zegura et al. found that the average response time of servers returned by anycast was significantly lower than random selection or the nearest. Not only was the response time lower but the variance was also lower, and in the case of random selection, by an order of magnitude.

Carter et al. [7] describe Manycast, its multi-recipient anycast, as something that "fills the spectrum of network communication space between anycast and multicast". The ability to send to multiple members of the group is particularly inviting as it allows multicast-esque abilities with the notion that a particular subset of nodes meeting a selection criteria will be contacted. For example, if a node wants to receive concurrent streams of the same video from similarly performing nodes in order to build up layers of the video, or to replicate data on servers with particular storage characteristics.

The motivation to employ a hierarchical design due to the benefits in resilience and resource utilization. Ganesan et al. [10] present the notion of hierarchical DHTs which have greater fault isolation, better bandwidth utilisation, hierarchical storage and access control. Garces-Erice et al. [11] state that hierarchical DHTs can "significantly" reduce the number of peer hops in a lookup and latency.

By providing locality aware n-casting capabilities, the overlay can provide applications the ability to contact the *k*-closest nodes for applications such as video broadcasting, file transfer or resource pooling.

We propose the use of hierarchical clustering [1] and hash sketches to create a multi-tier overlay that will allow nodes to maintain knowledge of group membership in a resource efficient manner.

2. Hierarchical Clustering

Linkage is the term given to the heuristic used to aggregate clusters. The linkage determines the characteristics of clusters and has a significant bearing on the performance an overlay that uses hierarchical clustering. To that end there are three common types of linkage that needs to be considered, single, complete and average link clustering.

In single link clustering clusters are aggregated based on the distance between the two closest nodes. Conversely, in complete-link clustering the clusters are merged based on the distance between the two furthest nodes in two clusters. With single-link clustering, the outliers in each cluster is not taken into account, meaning outliers can significantly alter the 'quality' of clustering, whereas complete-link, by acknowledging outliers, creates merges with the smallest possible diameter. Average-link clustering aggregates clusters by the mean distances within the cluster. There

Using the Peerwise dataset [9] to generate a dendogram using the Open Source Clustering algorithm [12] highlighted the problems of using complete-link clustering. Due to outliers, complete-link clustering resulted in nodes being included in clusters that would be better served being part of separate clusters. Single-link clustering would leave outliers in separate clusters and would only aggregate at higher levels, a behaviour that could lead to a rich club of clusters. Average-link clustering would, as its name suggests, provide a mid-point and aggregate clusters that are far-away at lower tiers.

3. Architecture

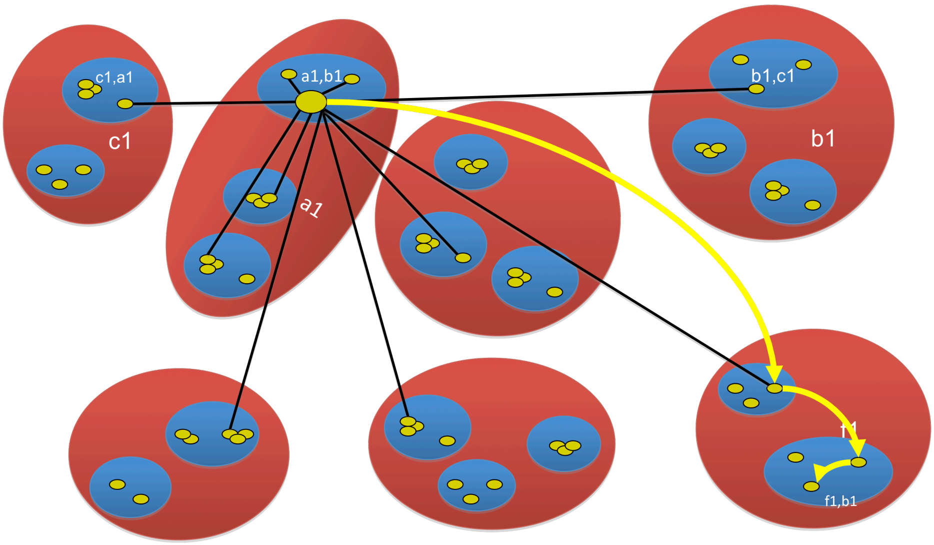

Figure 1- Structure of a hierarchical n-casting overlay

The clustering of nodes in Figure 1 is by their distance, which is represented by latency. The goal is to cluster nodes by geographical distance, meaning that nodes have extensive knowledge of geographically close nodes while retaining some knowledge of distant nodes.

Figure 1 shows a two-tier n-casting overlay where the smaller circles, such as *a1,b1* and *c1,a1*, represent a cluster in the lower tier and the larger circles, such as *a1* and *b1*, represent the upper tier. A node within a cluster maintains connections to all other nodes in that cluster, and one connection to other clusters in that tier, for example nodes' routing tables in cluster *a1,b1* would contain all three nodes in *a1,b1*. Each node also maintains a connection with one node in a distant cluster, for instance in Figure 1 that would be *c1,a1*.

When a node joins the overlay its proximity metric, which in our case would be latency, is used to place it in a cluster. Once part of the cluster, the node exchanges hash sketches, detailed in Section 4, to build up routing tables and group membership knowledge.

To query a node in a different cluster; one that is not in a node's routing table, the node first forwards the query to the node it knows is part of the higher tier cluster. The node in cluster *a1,b1* forwards the query to a node in cluster *f1,a1*. The node in cluster *f1,b2* then forwards the query to the node it has knowledge of in cluster *f1,b1*, where the node has knowledge of all nodes in the cluster *f1,b1*.

4. Efficient storage of group membership

N-casting introduces a new problem to anycast, the accurate knowledge of group membership. It is possible that as nodes report the membership of anycast groups, the receiving node attains duplicates and therefore has an 'inflated' view of group membership. While this problem exists is standard anycast, as the querying node is simply going to select a single host, rather than query *N* nodes. The result of this could be that a node sends out requests to *N*++ nodes when there are *N* nodes in a particular anycast group.

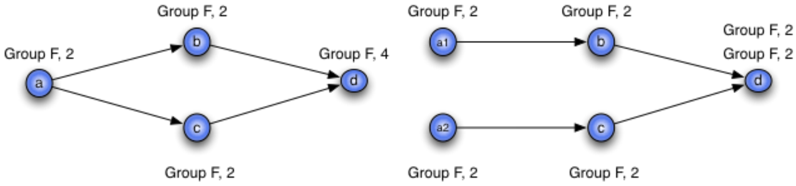

Figure 2 - Group membership double counting and non-aggregation

The issue of duplication shown on the left in Figure 2 can occur for two reasons, node d is unaware of membership data beyond one hop and that membership data is aggregated to reduce the size of nodes' routing tables. Nodes are typically unaware of membership beyond the previous hop because should a node have to transfer the full history of membership on each hop, the size of data transferred would become prohibitive with overlay membership growth.

Similarly, aggregation of groups is done for this reason as the number of groups increase without some means of data compression, the transfer of data, would grow exponentially with the number of groups, and result in a system that does not scale.

In order to overcome this we propose the use of distributed hash sketches [8]. It is a data structure that provides of fixed size and one that does not double count. Distributed hash sketches meet both the size and duplicity requirements of our n-casting system.

$$h(x) : \mathcal{D} \to [0, 1, \ldots, 2^{\lambda}] \qquad \rho(x) : [0, 1, \ldots, 2^{\lambda}] \to [0, 1, \ldots, \lambda]$$

Equation 1 – uniform hashing function (left) & least significant bit transformation (right)

The efficient use of hash sketches is based on the premise that objects from a multiset are mapped into integers using a uniform hashing function as presented in Equation 1. The resulting values are then mapped by a least significant bit function *ρ()* such that the transformation from *h(x)* can be ordered by the least significant bit. Using the least significant bit after *ρ()* allows us to use a cardinality estimator *R()*.

$$R(\mathcal{D}) = \max_{x \in \mathcal{D}} \rho(h(x)) \qquad C = 2^{R(\mathcal{D})}$$

Equation 2 - Cardinality estimator

In Equation 2, *C* has an estimation error of 1.87 binary orders of magnitude, however this can be reduced if *ρ()* is run multiple times on *h(x)* taking the arithmetic mean.

Hash sketches have the property that as the cardinality of multisets grows its accuracy reduces. While this may seem counterintuitive, for large sets it is not important for us to know the exact membership figure, just the upper bound to a certain degree of accuracy. As previously mentioned, accuracy can be improved if the *ρ()* function is repeated, increasing computational cost.

The design requirements put forward by an n-casting overlay is met by hash sketches that provides a data structure that is both scalable and maintains data integrity.

5. Conclusions and current work

Currently we are recording data that can be used to generate clusters that are representative of the Internet. Although we conducted preliminary clustering tests on the Peerwise data as mentioned in Section 2, the dataset has in the region of 1,200 nodes and we feel that a larger dataset is required to accurately represent real world conditions.

We believe that the fundamentals of this n-casting system with its distance-aware hierarchical clustering and its efficient hash-sketch data structure will result in a scalable overlay that provides applications with the N-closest nodes in a reliable manner.

References.

[1] A.K. Jain and R.C. Dubes. Algorithms for clustering data. 1988.

[2] D. Katabi and J. Wroclawski. A framework for scalable global IP-anycast (GIA), Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication ,October 2000

[3] T. Hardie. Rfc3258: Distributing authoritative name servers via shared unicast addresses. Internet

RFCs, 2002.

[4] C. Partridge, T. Mendez, and W. Milliken. Rfc 1546: host anycasting service, 1993.

[5] M. Castro, P. Druschel, A. M. Kermarrec, and A. Rowstron. Scalable application-level anycast

for highly dynamic groups. Group Communications and Charges, 2003.

[6] Zegura, E.W., Ammar, M.H., Zongming Fei, Bhattacharjee, S., "Application-layer anycasting: a server selection architecture and use in a replicated Web service," *Networking, IEEE/ACM Transactions on* , vol.8, no.4, pp.455-466, Aug 2000

[7] Casey Carter, Seung Yi, Prashant Ratanchandani, and Robin Kravets. Manycast: exploring the space between anycast and multicast in ad hoc networks. In MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking, pages 273–285, New York, NY, USA, 2003. ACM.

[8] N. Ntarmos, P. Triantafillou, and G. Weikum. 2009. Distributed hash sketches: Scalable, efficient, and accurate cardinality estimation for distributed multisets. *ACM Trans. Comput. Syst.* 27, 1, Article 2 (February 2009)[9] C. Lumezanu, D. Levin, and N. Spring. PeerWise discovery and negotiation of faster paths. In Proc. Workshop on Hot Topics in Networks (HotNets). Citeseer, 2007.

[10] P. Ganesan, K. Gummadi, and H. Garcia-Molina. Canon in g major: designing dhts with hierarchical structure. In Distributed Computing Systems, 2004. Proceedings. 24th International Conference on, pages 263–272, 2004.

[11] L. Garces-Erice, E. W. Biersack, P. A. Felber, K. W. Ross, and G. Urvoy-Keller. Hierarchical peer-to-peer systems. Lecture Notes in Computer Science, 2003:1230–1239, 2003.

[12] M. J. L. de Hoon, S. Imoto, J. Nolan, and S. Miyano: Open Source Clustering Software. Bioinformatics, 20 (9): 1453--1454 (2004)