ROGS: An Optimal Location Tree Construction Technique for Content Oriented Networks

Sai Ganesh Sitharaman
Department of Computer Science
Texas A&M University
College Station, TX 77840
ssai@cs.tamu.edu

ABSTRACT

We propose an optimal location tree construction technique that takes into account the resilience of nodes and accommodates at most *k*-1 link failures. We define a 2-dimensional collection of vertices in a graph in which vertices of each dimension have path from one to another. A simple graph matching algorithm between these *dimension vertices* ensure the connectivity with all other nodes in the network. We use *strongly connected graph components* to build superposition of 2-trees that ensure optimal resilience of paths. The number of routing entries are fixed per node and are the only ones used to forward among any members in the network. We do not have any simulation results supporting the theory but planning to work on it soon.

1 INTRODUCTION

Computing and communication are considered synonymous in the current information age. Attempts are underway towards development of network operating system and ubiquitous computing. Network infrastructure as early as 1999 helped to create peer-to-peer computing environment. The novel *SETI@Home* project initiated by Space Sciences Institute of University of California, Berkeley during 1999, initiated a massive use of idle distributed computing resources to search for intelligible patterns in the radio signals of extraterrestrial origin. The first peer-to-peer network primarily relied on end host application to download large data files and search for patterns. The strength of such a massive scale distributed application is now recently being realized in a new form of information exchange paradigm in which large and large amount of information exchange and technology to allow such information exchange possible.

Peer-to-peer distributed sharing system such as Napster and Gnutella are the primary movers of such high degree of end-to-end host content. Estimates show that Gnutella shared about 3 billion music and video files tallying about 6.1 terabytes of data among the users during August 2001. In a related article by Brownlee and Claffy [1], members of CAIDA group, suggested that 40-70% of the Internet traffic streams are *dragonflies* lasting for about 2 seconds or less and about 1.5% are long-running *tortoise* streams lasting about 15 minutes. In their experiments, it is these *tortoises* that contribute up to 50% of torrent bytes of data in the Internet. Regardless of the actual statistics, we can infer that the Internet traffic is moving towards a heavily content oriented network, from simply sharing music between end-to-end hosts to efficient content distribution. The question immediately arises in our mind whether the well understood dynamics and ubiquity of the Internet is indeed good enough to support such requirements. In this article, we focus on the underlying network technologies that are extension of original Internet design.

Section 2 gives some background on the problems in existing approaches particularly with multicast. The section also introduces various types of overlay and content distribution networks that extend multicast. Section 3 proposes a new location based tree construction scheme *ROGS* and gives algorithms to achieve resilience. Section 4 concludes the paper with problems and scope for future work. Section 5 gives many related references.

2 BACKGROUND AND RELATED WORK

IP multicast architecture was primarily introduced to increase efficiency of data delivery. Multicast-enabled routers need to keep routing and group membership state information in edge routers to transparently serve heterogeneous members. We have learnt earlier that IP multicast is plagued with several problems, despite being a simple service. IP multicast exhibits several technical overheads such as: scalability and heterogeneity of members, flow and congestion control, achievement of receiver synchronization, determining fairness among unicast and multicast flows, and consistency of multicast implementations among service providers. Thus, an alternative mechanism is to be found to accommodate the new age information exchange services among users.

Overlay network is an architecture that makes use of traditional unicast path between end systems instead of multicast. It is so named because it lies as application on top of one of more networks and provides an additional layer of indirection to existing services. Overlay networks change the properties of the underlying IP network by seamlessly providing application level services instead of relying upon network layer services. An overlay network may be fully distributed and high self-organizing network that does not require any more service other than traditional IP unicast. It is thus realized that such networks can provide significantly good performance by moving routing and group membership overhead to end hosts rather than routers.

Paxson [2] showed that persistence of Internet route and route stability is something to be considered while designing newer applications. In a massive *traceroute* experiment with about 860 source/destination pairs, Paxson demonstrates that more than half the routes are short-lived and are less than a week. Moreover, routes that persisted more than a week accounted for 90% of total persistence. Any new service thus needs to accommodate this dynamics of the Internet. One immediate inference is that service overhead should be pushed to higher layers and overlay networks is one form of such service.

Overlay networks are highly susceptible to network changes, as end users are often not graceful enough to the network and to the other users. This results in network instability and random route oscillations. Several solutions are possible to remedy these including clustering based on geographical awareness and caching techniques. Peers fail often and have to be replaced with an alternative-serving node, preferably in the same network or better. Thus, it becomes important to locate nodes based on cache indices of objects present in the failed node. Typical techniques include forming a hash function that maps objects to corresponding node identifiers that may have these objects. A failed peer has to replicate or transfer complete state information to another that can claim it, but hashing function has to be announced again to the network.

Chu *et al.[3]* propose a novel end system multicast overlay network that moves group membership and flow control issues to unicast. They present *Narada* protocol constructing self-organizing overlay network a mesh construction followed by spanning tree. Mesh construction is more resilient to failures. Members maintain topology and group information of all members. Performance is measured by link stress, *resource usage,* and *relative delay penalty (RDP).* Narada protocol handles join, leave and failures. Simulations suggest 90% of members have good RDP. However, the protocol overhead increases linearly with group size.

Krishnamurthy and Wang [4]propose clustering for closer content replication using BGP routing and longest prefix matches. The CIDR-based approach performs two-step process: prefix extraction and cluster identification with longest prefix match. Experiment with live Internet route tables suggests more than 95% clients containing less than 100 clients. About 1% of total clusters issue significant number of web requests and the distribution is heavy-tailed. Cluster identification may be misled when clients not in topologically closers are in same cluster and when other clusters contain clients that belong to the cluster in question. Validation is done using *nslookup* and *traceroute*.

Jannotti *et al.* [5] introduce a single-source overlay network system that creates high-bandwidth paths from the root to all nodes without sacrificing bandwidth. Global overall status of distribution tree is maintained at the root and all changes are propagated upwards. In the three step process, the protocol does initialization, tree building and tree maintenance. Tree building proceeds placing a node as far away from the root without sacrificing bandwidth, thus forming a "deep distribution tree". Node periodically reevaluates its position with parent, grandparent and siblings. Periodic birth and death certificates and changes to nodes are issued to parents. Race condition exists between issuance of birth/death certificates. Sibling optimization can lead to oscillations.

Chu *et al.* [6] extend end system multicast proposed in [3] to enable conferencing type of applications by explicitly considering bandwidth and latency. Optimization is done for dual metrics but bandwidth is given higher priority to latency. Basic Narada protocol is extended to build the new bandwidth-latency based overlays and compared with sequential unicast. Most hosts sustain 95% source rate on average and achieve latencies comparable to unicast. With a wider network, bandwidth-latency model provided 50% higher throughput than latency-only. The protocol itself added network overhead of 10-15% for small groups.

Ratnasamy *et al.*[7] study a novel content-addressable network (CAN) that is based on spatial orientation of nodes and corresponding hash function mapping. A *d*-dimensional Cartesian coordinate space is dynamically partitioned and zones assigned to nodes. Key *K* maps uniquely to point *P* in the zone of that node. Node joins and leaves result in zone splitting and merging respectively. Each node maintains its neighbors and routes using them. Average routing path length is O($n^{1/d}$) for 2*d* neighbors. Periodic soft state refresh messages ensure faster route convergence. Typical problem in this model is the maintenance of at most *n–1* neighbors in the worst case.

Stoica *et al.* [8] propose a novel decentralized key and value lookup service called *Chord* that identifies the node storing the key. Chord uses consistent hash function and uses the uniformity of hash results to scale *1/N* fraction of keys, where *N* is the total nodes in the network. For *m*-bit input key identifier, nodes maintain *m* routing entries in *finger table*. Node joins initialize predecessor, update finger table of existing nodes and copies keys. Updating fingers results in an average of O(*1/N*) moving of key-value pairs to a different location. Each node maintains only O(log *N*) information about other nodes and updates require O($\log^2 N$) messages.

Liebeherr and Nahas [9] study a geometrical layout of overlay network topology using delaunay triangulation. Each node in such a topology generally has a smaller edge degree. Triangulation has a set of alternative non-overlapping routes between pairs that make them resilient. Forwarding is done using spanning trees and geometrical compass routing is used to locate neighbors. Further improvements are performed with hierarchical triangulations with special responsibilities for higher nodes. Such topological schemes do not require any measurements in constructing the overlays.

Krishnamurthy *et al.* [10] study the performance of content distribution networks (CDN) and how requests are offloaded from origin servers and impact of client-perceived latency. DNS redirection and URL rewriting is primarily used to study client performance. The study analyses the variation in the server selected for individual clients and total servers available, since the specific policies and algorithms are proprietary. Results show that CDNs perform better with fixed servers.

Ng and Zhang [11] demonstrate yet another coordinate reference-based approach to predict optimum host distance. Such metric may be effectively used for making overlays topologically aware. A number of reference *landmarks* are distributed across the Internet and all hosts compute relative distance with each of these landmarks. A combined error function is then minimized to determine the degree of closeness of nodes. For a *D*-dimension space, the number of landmarks must be at least D. Additional landmarks may provide effective use until a certain threshold. The techniques perform twice as better as other approaches.

Banerjee *et al.* [12]investigate a novel low-bandwidth hierarchical clustering application-layer multicast peers named NICE. NICE builds clusters of nodes *nearby* and maintains levels of clusters of O(log *N*) members, where *N* is the number of nodes in the tree. Each cluster is small and lies between *k* to 3*k*-1, for some constant *k*. Clusters have leaders and leaders themselves form clusters at higher layers. At the top layer, the single root cluster is responsible for changes in the distribution tree. Control and data path between nodes are different. Worst case message overhead is O(*k*log *N*).

Chawathe *et al.* [14] proposes a reliable hybrid split unicast-multicast model that uses TCP like reliable protocol for unicast and retains efficiency of IP multicast. Large multicast sessions are divided into smaller data groups that contain homogeneous partners. An application aware agents (RMX) use the application semantics to adapt heterogeneity. The model relies on end-to-end loss recovery mechanism, and application naming technique to transport data through RMX. Data delivery is through scattercast forwarding. A naming protocol preserves application specified data across flows.

Shaikh *et al.* [15] presents experiments performed using DNS-based servers to determine the impact of cache-less DNS and identifies client location and performance using redirected DNS servers. The authors identify client latency due to the incorrect DNS redirection and also increase in latency due to embedded objects in page. Using HTTP and DNS request logs, they study the client nameserver proximity to the DNS server and analyzes the latency effect of clients.

Banerjee *et al.* [16] consider the problem of application peer finding that can be used in a peer-to-peer system. End systems organize themselves into clusters independently without infrastructural support. Clusters are further organized hierarchically and each peer has limited knowledge of other peers in the same group. Cluster leader maintain O(log *N*) state, where *N* is the number of peers. Accuracy, query latency and control overhead are studied in the article.

Chawathe and Seshadri [19] propose architecture for interconnecting several broadcast networks into a loosely coupled broadcast systems. The protocol involves routing protocol, tree-building protocol and data forwarding protocol. Federation consists of broadcast gateways and peering links. Federation trees are built with a shared inter-BN distribution trees rooted at the owner. Joining nodes initiate session specific route discovery and response messages. Data forwarding layer abstracts UDP/TCP messages exchanged.

Zhao *et al.* [17] propose a novel overlay location and routing infrastructure providing location-independent to the closest copy. Node insertion operation occurs with population of neighbor maps and informing neighbors. Soft state republishing is used to update location pointers. Tapestry allows control the location of data allowing systems to manipulated with references.

Rowstron and Druschel [18] propose an alternative location based architecture that uses logarithmic number of steps to locate an object. Nodes are identified with identifiers and search is done by locating the object closest to the given key. Each input is scanned for a constant number *c* digits. Pastry nodes maintain state information of neighborhood and leaf set. Pastry takes into account network locality when routing messages.

3 ROGS: A RESILENT PEER-TO-PEER SYSTEM

In this section, we present an optimal resilient graph theoretic-based peer-to-peer model that makes use of a well-known property of Depth First Search (DFS) algorithm. We construct distinct set of disjoint location trees that by themselves are linear search trees but when combined yield an optimal structure for routing. ROGS system is $k\text{–}1$ resilient, for constant k neighbors of any given node. This means, all but one neighbor fails but is still reachable. Moreover, each node only keep a constant number of routing entries and this makes the scheme scalable.

Section 3.1 presents background and motivation in developing the scheme. Section 3.2 describes the strongly connected component graph problem and the superposition trees of our scheme. Section 3.3 describes our tree building approach.

3.1 Background and Motivation

Motivation for this approach developed from analyzing Pastry forwarding structure. Given a message and key K, Pastry routes a message efficiently in O(log N) steps, where N is the number of nodes in Pastry network, with a node identifier that closely matches the key K in question. Logarithmic routing efficiency is achieved at each node by having $\log_c N$, where c is the number of entries (columns) for each row corresponding to various possible node identifier digits. Each node however maintains a variable number of routing entries and routes according to this matrix.

$$\text{Entries in Pastry routing matrix} = c * log_c N \qquad (1.\ )$$

With a value of $c=15$ and $N=10^6$, the total routing entries is thus the total number of entries is 76. Although this looks a manageable number, it is not efficient in managing failures of any type. In the event of malicious or failed node, the client repeats the query a number of times before it is being detected. Pastry is self-organizing, so control information exchanged exclusively during routing failures or partition is significant and even can take longer time before route stabilizes. It is thus motivating to think of a resilient routing structure that however preserves the logarithmic routing efficiency. Although not trivial, it would be encouraging to verify for constant number of route entries per node in the overlay network.

3.2 Strongly Connected Component and Superposition trees

A directed graph is strongly connected if there exists an unique path from one vertex to another and vice versa, for all such vertices in the graph. *Strongly connected components (SCC)* are equivalence classes of vertices under this mutually reachable relation [13]. SCC are an application of classic depth-first search in which a directed graph is decomposed in multiple strongly connected components. Each such component is rooted at a particular vertex and has path from every vertex to every other vertex in that tree. The result of the decomposition is thus a set of disjoint trees each of which is strongly connected by itself. Typical SCC finding algorithms perform two depth-first searches – one on the original graph and the other on original graph transposed.

Strongly connected components gives us useful clues to establish resilience in a general graph. Depth-first search on a given graph gives us multiple disjoint SCCs, each rooted at different vertices. In other words, our final graph structure must establish strongly connected behavior after several vertex or edge failures.

One solution to the problem would be to consider these several disjoint trees having the same vertices but the edge in any one tree is never available in any other tree. Note that this is strictly not a property of SCC. We build superposition trees using this property.

Several optimal tree construction algorithms are discussed in literature [13] and are not discussed here. One thing that is to be observed for effective resilience is that, we require the in and out degrees of vertices be the same, when all these trees are superimposed to form a single mesh. Our model constructs such a tree and demonstrates that it is *k-1* failure resistant.

3.3 ROGS tree construction

We now describe the construction of such superimposed tree structure for ROGS peer-to-peer system.

We define a dimension *D* of the system to be a logical collection of related nodes in the network. Collection can be extended to any sort such as geographical or having no physical significance such as some number theory property of node identifiers. We thus have *D*-independent collection of node identifiers. We design our algorithm with *D=2* and the significance of this will be known later. It may however be extended to other values of D.

We simply use the number property of node identifiers as collections in each dimension. We construct two intermediate graph G_e and G_o, where G_e is the collection of even numbered vertices and Go is the collection of odd numbered vertices. Each vertex in graphs G_e and G_o have an undirected path to any other vertex in that graph. There is an edge from nodes of increasing order.

Figure 1 below demonstrates a simple linear tree.

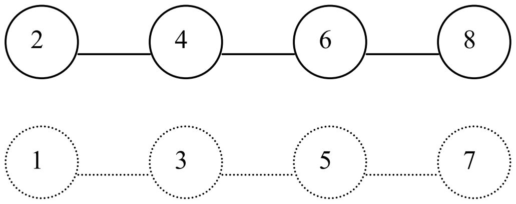

Figure 1. Simple linear tree construction

Next, we construct a simple matching graph between these two G_e and G_o as follows. For each node identifier n_e in G_e, connect it to $n_e/2–1($mod $n)$ and $n_e/2+1$ (mod n) in Go, where n is the total nodes in the network. This results in the following connection for the above graph.

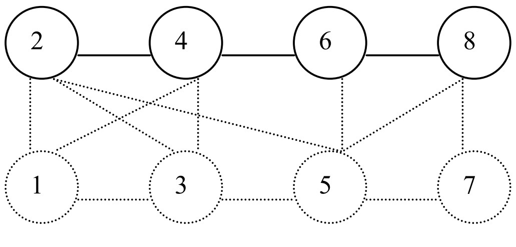

Figure 2. Complete routing tree construction with 2 superimposed trees

Routing in such a network is made effective in O(log *n*) operations by allowing forwarding over edges in both Ge and Go in final G. For instance, the upper bound of the above described graph if between 1 → 8 with 3 hops. At the same time, routing entries are kept constant at O(log *n*) for each node.

Resilience in this network is k-1 possible, meaning that *k-1* neighbors can be disconnected but the node still have forwarding to all other reachable nodes.

We plan to study the basic operations of add, delete and updating node identifiers and compare it with other schemes.

Pseudo-code for ROGS tree construction.

```
1. for each vertex $v$ in Ge with node identifier $n_e$
2.     Construct link to $G_o$ left = $n_e/2 – 1($mod $n);$
3.     Construct link to $G_o$ right=$n_e/2 + 1($mod $n);$
4. done
```

For the simple graph in figure 1, we show the newly constructed one in figure 2.

4 CONCLUSION

Overlay networks shifts multicast support and overhead from routers to end systems. Moving such services to end systems help in two fold: in the construction of useful services and make such services adaptive to the network dynamics. Thus, overlay networks are one form of self-organizing and self-improving protocols that does not rely on any special support from intermediate routers. Such services can be made to dynamically adapt to network congestion and failures.

We learnt that overlay networks can be broadly classified as: tree-first approaches and mesh-first approaches. Narada is an example of mesh-first and NICE is an example of tree construction. Several articles address network as some form of geometrical distribution such as CAN. Other forms consider overlays as distributed hash-based approaches such as Chord and Pastry.

Finally, we presented our own theoretical approach ROGS peer-to-peer system towards building distributed tree structures to optimize route forwarding and that is resilient against failures. The scheme is yet to be tested for correctness using simulations and scalability factor is to be analyzed.

ACKNOWLEDGEMENTS

The author likes to thank the IRL team at Texas A&M University and Dmitri Loguinov for bringing in the problems to context and helping identify the potential pitfalls of various approaches.

5 REFERENCES

[1] Nevil Brownlee, KC Claffy, Understanding Internet Traffic Streams: Dragonflies and Tortoises, IEEE Communications Magazine, Vol. 40, No. 10, Oct 2002.

[2] V. Paxson, End-to-End Routing Behavior in the Internet, IEEE/ACM Transactions on Networking, Vol.5, No.5, pp. 601-615, October 1997.

[3] Y. Chu, S.G. Rao, and H. Zhang, "A Case for End System Multicast," ACM SIGMETRICS, June 2000.

[4] B. Krishnamurthy and J. Wang, "On Network-Aware Clustering of Web Clients," *ACM SIGCOMM*, 2000.

[5] J. Jannotti, D.K. Gifford, and K.L. Johnson, "Overcast: Reliable Multicasting with an Overlay Network," *Symposium on Operating Systems Design and Implementation (OSDI)*, October 2000.

[6] Y. Chu, S.G. Rao, S. Seshan, and H. Zhang, "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture," *ACM SIGCOMM*, 2001.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network," *ACM SIGCOMM*, August 2001.

[8] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balarkishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *ACM SIGCOMM*, August 2001.

[9] J. Liebeherr and M. Nahas, "Application-Layer Multicast with Delaunay Triangulations," *IEEE GLOBECOM*, November 2001.

[10] B. Krishnamurphy, C. Wills, and Y. Zhang, "On the Use and Performance of Content Distribution Networks," *ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[11] T.S.E. Ng and H. Zhang, "Predicting Internet Network Distance with Coordinates- Based Approaches," *IEEE INFOCOM*, 2002.

[12] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," *ACM SIGCOMM*, August 2002.

[13] T.Cormen, C.Leiserson, R.Rivest, Introduction to Algorithms, MIT Press, Cambridge, MA, Edition 1990.

[14] Y. Chawathe, S. McCanne, and E. Brewer, "RMX: Reliable Multicast for Heterogeneous Networks," *IEEE INFOCOM*, March 2000.

[15] A. Shaikh, R. Tewari, and M. Agrawal, "On the Effectiveness of DNS-based Server Selection," *IEEE INFOCOM*, April 2001.

[16] S. Banerjee, C. Kommareddy, and B. Bhattacharjee, "Scalable Peer Finding on the Internet," *IEEE GLOBECOM*, 2002.

[17] Y. Zhao, J.D. Kubiatowicz, and A. Joseph, "Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing," *UC Berkeley Technical Report UCB/CSD-01-1141*, April 2000.

[18] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems," *IFIP/ACM International Conference on Distributed Systems Platforms*, November 2001.

[19] Y. Chawathe and M. Seshadri, "Broadcast Federation: An Application-Layer Broadcast Internetwork," *ACM NOSSDAV*, 2002.