Unstructured Peer-to-Peer Networks - Next Generation of Performance and Reliability

William Acosta Surendar Chandra
University of Notre Dame
{wacosta, surendar}@cse.nd.edu

ABSTRACT

In this poster we will present our work on the design of efficient and reliable unstructured peer-to-peer (P2P) systems. Our work focuses on creating well-connected unstructured P2P overlay that can perform efficient searching and message routing. We show that well designed systems can tolerate high node failures (>25%) while maintaining connectivity and still resolving searches with few messages. This is important in applications such as file sharing and content distribution where there are many thousands of participating nodes that are widely dispersed and network coniditions are highly variable. Structured P2P systems are not suitable for these applications since such applications require multi-attribute and wild card searching. We show that carefully constructed overlays can resolve this type of search within 4 hops for large networks ($>$ 10,000 nodes) with low object replication ratios ($<$ 1%).

1. INTRODUCTION

Peer-to-peer (P2P) file sharing networks such as Gnutella [2], Kazaa [3] and BitTorrent [1] have become increasingly popular. The popularity of P2P networks has fueled interest in leveraging them to build large scale distributed applications such as distributed data storage [10], cooperative backup [8], and distributed multicast [6].

Consider the problem of distributing large ($>$ 10 MB) multimedia files across a wide area network to many users. A centralized approach requires the publisher of the content to have the server and network infrastructure to host the content. This requires enough bandwidth to handle a large number of users simultaneously downloading the content. Such infrastructure does not scale well as the number of users increases and requires a large financial commitment that small content publishers will likely not be able to make. Additionally, such centralized infrastructures are susceptible to failure as well as targeted attacks. Ideally, the content publisher would place a few replicas of the content on different nodes in the network and users would get redirected to the replica that is nearest to them. If the object becomes popular, more replicas of it should be created to limit the bandwidth consumption on wide area network links by allowing nodes to download replicas that are near them. On the user side of the application, objects and replicas in the system need to be discovered efficiently. This includes efficiently locating the nearest replica. The decentralized and self-organizing nature of P2P networks are ideally suited to solving this type of problem.

In this poster we will present our work exploring the use of unstructured P2P systems as viable platforms for distributed sharing and distribution of content. Specifically, we will present our work in exploiting the inherent flexibility of the unstructured P2P model to create well connected and fault-tolerant overlays with efficient searching and routing mechanisms. Our work shows that these efficient P2P networks can significantly outperform current unstructured P2P networks by overcoming the key limitations of these systems.

2. UNSTRUCTURED P2P SYSTEMS

Unstructured P2P overlays are inherently flexible in their neighbor selection and routing mechanisms. They can leverage proximity information of the underlying network to localize the communication pattern of the system. They can create topologies that are resilient to random node failures as well as withstand targeted malicious attacks. However, traditional unstructured P2P overlays do not exploit many of these benefits. Analysis of popular unstructured P2P networks shows that current systems create topologies [12] and utilize search mechanisms [9] that do not match the underlying network characteristics. In particular, these systems exhibit preferential connection tendencies toward highly connected nodes as witnessed in power-law networks [4]. Such overlays are vulnerable to node failures of these highly connected nodes. Additionally, these overlays exhibit high communication costs since the peer selection process ignores network proximity and thus nodes tend to select neighbors that are distant in terms of network latency.

Creating an unstructured P2P topology with desirable proximity awareness properties poses several problems. First, transient network conditions and node lifespans require a distributed solution where each node makes independent decisions with limited dependence on global information from other nodes. Also, some nodes may naturally appear desirable to many nodes; the topology generation mechanism should balance the use of proximity information with the capacity of the node to service these neighbors. The topology should also maintain global connectivity, especially in the face of node failures. Our aim is to examine the performance of the different algorithms used for peer selection in unstructured P2P systems and determine which algorithms yield overlays that best achieve the goals of low communication cost, good connectivity, and efficient searching.

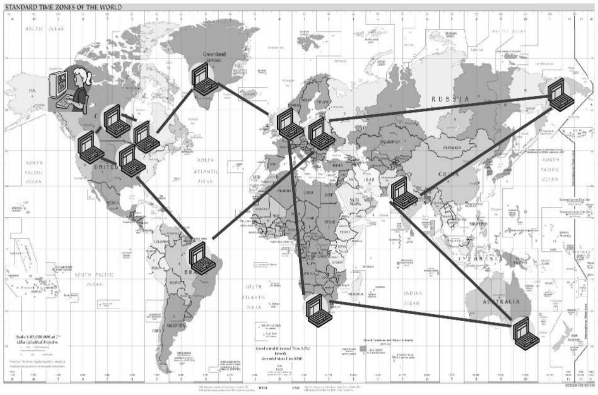

Figure 1: Example of an overlay with a ring topology over a wide area network.

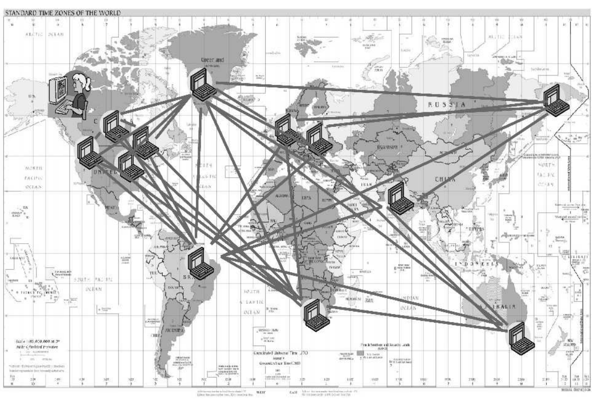

Figure 2: Example of a fully connected overlay topology over a wide area network.

3. CONNECTIVITY PROPERTIES OF P2P OVERLAYS

We are interested in determining whether a given overlay has desirable connectivity properties. The connectivity and compactness of the overlay affect the fault-tolerance of the overlay to node failures as well as the ability to reach many nodes quickly and thus affecting the efficiency of search mechanisms. However, there is a need to balance good connectivity with network scalability As an example, consider a ring topology as shown in Figure 1. Ring topologies are sparse ($O(n)$ edges), so each node only needs to maintain two connections. However, the network can be easily partitioned with just two node failures. Further, because each node maintains only two connections, high capacity nodes will be underutilized. On the other hand, a fully connected topology as shown in Figure 2 can tolerate many faults before the network becomes partitioned. However, because such a network is dense and has many edges ($O(n^2)$ edges), each node must maintain many connections. This solution is not scalable. It also forces low capacity nodes to maintain more connections than they may be able to handle.

3.1 Algorithms for Unstructured P2P Systems

The process of creating and maintaining the overlay in P2P systems is decentralized and distributed. Each node must make local decisions without requiring each node to have global information about the system. To create the overlay, nodes find peers that are already in the network, they then evaluate which peers would be better neighbors, and then connect to those peers. We can describe a P2P system with the following abstract algorithm:

repeat
  Locate Candidate Set of Peers
  Evaluate(Candidate Set)
  Select as neighbors the "best" peers
until No more neighbors needed

The details of each of these operations are dependent on the actual P2P implementation. However, the key to the algorithm is the *Evaluate()* function since this function determines which nodes will become peers in the overlay. Because unstructured P2P systems do not use a deterministic global algorithm for selecting the appropriate neighbors, the design of the *Evaluate()* function significantly affects the performance and scalability of the system. This function needs to operate without global knowledge of the system. For our work, we created a simulator and implemented various algorithms as implementations of the *Evaluate()* function described above. These algorithms are described below:

High-Degree Biased (Power Law) In this strategy, nodes are biased towards selecting peers with high degree (many neighbors). Gnutella implements this approach [13].

Proximity Biased A proximity biased evaluation function will measure the network latency between nodes and select as peers those nodes that are closer in terms of network distance.

Connectivity Biased This evaluation function is biased towards nodes that give access to nodes in the network that were not previously accessible through the current connections.

Proximity-Connectivity Hybrid The proximity-connectivity hybrid algorithm expands the evaluation function of the connectivity algorithm to also take into account proximity.

In the poster we will present our results of the examination of the overlay topology generated by the unstructured P2P algorithms described above. Specifically, we will present comparisons of the topologies in terms of connectivity, communication cost, and tolerance to node failures.

4. SEARCH IN UNSTRUCTURED P2P NETWORKS

Unstructured P2P systems do not rely on the overlay itself for routing; they have no inherent restrictions on the type of searches that they can perform. For example, a flooding search can be used for either identifier search or wild-card and attribute search. Unless care is taken when building the overlay, flooding can be inefficient. Identifier-based search where the identifier of the desired object is known in advance can also be solved in unstructured P2P if the overlay is constructed carefully. We can use an algorithm that leverages information at each node to probabilistically route the search message. In the following sections we discuss the details of flooding searches, random walks, and exact-match identifier searches in unstructured P2P systems.

4.1 Flooding Searches

In a flooding search, when a node receives a query, it simply forwards the query to all of its neighbors. However, [13] noted that

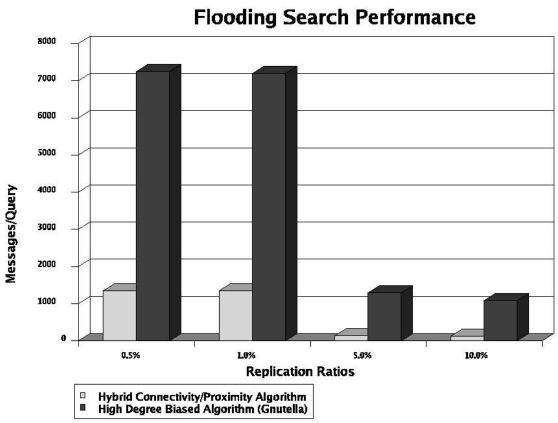

Figure 3: Comparison of flooding search performance under various object replication ratios for a traditional Gnutella network and one created using our hybrid connectivity-proximity algorithm. Our hybrid algorithm consistently requires fewer messages.

flooding in unstructured P2P systems such as Gnutella is inefficient. A query in Gnutella can overwhelm the network with messages. The traditional mechanism to bound the number of messages sent is to restrict the distance a query can travel in the network by using a Time To Live (TTL) in the query. The problem then is to find an appropriate TTL that can balance search success with minimizing messages. However, the inefficiencies of flooding are due largely to the fact that current unstructured P2P overlays are not designed with the search mechanism in mind. We assert that flooding can be efficient if the P2P overlay is constructed in such a way as to allow the broadcast nature of the flood to exploit the network structure itself to reach as many nodes as possible.

4.2 Random Walks

Under certain conditions such as when network bandwidth between peers is limited, it is necessary to sacrifice the response time for resolving a query in order to reduce the number of messages sent. A common approach is to replace a flooding search with a random walker [7]. A random walker, as the name implies, randomly walks the network querying each node it visits for the desired object.

4.3 Identifier Search

If the exact identifier of the desired object is known in advance, we can perform optimizations to the search mechanism in order to make it more efficient at locating the desired object. The approach we take to solve this problem uses attenuated Bloom filters [11]. A Bloom filter is a compact representation of a large set of objects that allows one to easily test whether a given object is a member of that set [5]. An attenuated Bloom filter is a hierarchy of Bloom filters, each of which contains aggregate information about some set of nodes. because the Bloom filters are essentially a potential function that guides the walk thus allowing the search to converge toward the object faster than a random walk while generating very few messages for each search.

In the poster we will present our findings on the performance of each of these search mechanisms under various replication ratios (0.5 % to 10.0%). For example, in Figure 3, we compare the number of messgaes required to resolve a search in a traditional Gnutella topology and in a topology created with a hybrid connectivity-proximity algorithm under various object replication rates. We will also show the performance of the search mechanisms in the presence of node failures on various overlay topologies.

5. REFERENCES

[1] Bittorrent protocol specification. http://bitconjurer.org/BitTorrent/protocol.html.

[2] The gnutella protocol specification v0.4. http://dss.clip2.com/GnutellaProtocol04.pdf.

[3] Kazaa media desktop. http://www.kazaa.com/us/index.htm.

[4] R. A. A.-L. Barabasi and H. Jeong. Mean-field theory for scale-free random networks. *Physica A*, (272):173–187, 1999.

[5] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 298–313. ACM Press, 2003.

[7] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 177–190. ACM Press, 2002.

[8] L. P. Cox and B. D. Noble. Samsara: honor among thieves in peer-to-peer storage. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 120–132. ACM Press, 2003.

[9] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 314–329. ACM Press, 2003.

[10] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells, and B. Zhao. Oceanstore: an architecture for global-scale persistent storage. In *Proceedings of the ninth international conference on Architectural support for programming languages and operating systems*, pages 190–201. ACM Press, 2000.

[11] S. C. Rhea and J. Kubiatowicz. Probilistic location and routing. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, June 2002.

[12] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6(1), 2002.

[13] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.