Efficient Gnutella-like P2P Overlay Construction

Yunhao Liu[1], Li Xiao[2], Lionel M. Ni[1] and Baijian Yang[3]

[1] Department of Computer Science, Hong Kong University of Science and Technology, Kowloon, Hong Kong, China
ni@cs.ust.hk
[2] Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA
lxiao@cse.msu.edu
[3] Department of Industry and Technology, Ball State University, Muncie, IN 47306, USA
byang@bsu.edu

Abstract. Without assuming any knowledge of the underlying physical topology, the conventional P2P mechanisms are designed to randomly choose logical neighbors, causing a serious topology mismatch problem between the P2P overlay network and the underlying physical network. This mismatch problem incurs a great stress in the Internet infrastructure and adversely restraints the performance gains from the various search or routing techniques. In order to alleviate the mismatch problem, reduce the unnecessary traffic and response time, we propose two schemes, namely, location-aware topology matching (LTM) and scalable bipartite overlay (SBO) techniques. Both LTM and SBO achieve the above goals without bringing any noticeable extra overheads. More-over, both techniques are scalable because the P2P over-lay networks are constructed in a fully distributed manner where global knowledge of the network is not necessary. This paper demonstrates the effectiveness of LTM and SBO, and compares the performance of these two approaches through simulation studies.

1 Introduction

As an emerging model of communication and computation, peer-to-peer systems are currently under intensive study [6, 10, 12, 15, 16]. This paper focuses on unstructured P2P systems, such as Gnutella [2] and KaZaA [4], since they are most commonly used in today's Internet. File placement is random in these systems, which has no correlation with the network topology. The typical search mechanism adopted will blindly “flood" a query to the network among peers (such as in Gnutella) or among super nodes (such as in KaZaA). The query is broadcasted and relayed until a certain criterion is satisfied. If an inquired peer can provide the requested object, a response message will be sent back to the source peer along the inverse of the query path. The flood mechanism ensures that the query messages can reach as many peers as possible within a short period of time in a P2P overlay network.

Studies in [15] and [14] have indicated that P2P systems, such as FastTrack (including KaZaA and Grokster) [1], Gnutella, and DirectConnect, contribute the largest portion of the Internet traffic. Among those P2P traffic, a considerable portion of the

H. Jin et al. (Eds.): NPC 2004, LNCS 3222, pp. 146-153, 2004.

load is caused by the inefficient overlay topology and the blind flooding, which also makes the unstructured P2P systems far from being scalable [13].

Aiming at alleviating the mismatch problem, reducing the unnecessary traffic, and addressing the limits of existing solutions, we propose location-aware topology matching (LTM) and scalable bipartite overlay (SBO) scheme. In LTM, each peer issues a detector in a small region so that the peers receiving the detector can record relative delay information. Based on the delay information, a receiver can detect and cut most of the inefficient and redundant logical links, and add closer nodes as its direct neighbors. SBO takes another approach where Gnutella-like peer-to-peer overlays are optimized by disconnecting redundant connections and choosing physically closer nodes as logical neighbors. Our simulation studies reveal that the total traffic and response time of the queries can be significantly reduced by both LTM and SBO without shrinking the search scope.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 discusses unnecessary traffic and topology mismatch problems. Section 4 outlines the designs of LTM and SBO schemes. Simulation and performance evaluation of the LTM and SBO are presented in Section 5, and we conclude our work in Section 7.

2 Related Work

Many efforts have been made to avoid the large volume of unnecessary traffic incurred by the flooding-based search in decentralized unstructured P2P systems. In general, three types of approaches have been proposed to improve search efficiency in unstructured P2P systems: forwarding-based, cache-based and overlay optimization. The above three different approaches are not exclusive and can be integrated to achieve better results.

In forwarding-based approaches, instead of passing on the query messages to all but incoming logical neighbors, a peer selects a subset of its neighbors to relay the query. The second approach is cache-based search, which includes data index caching and content caching. Centralized P2P systems provide centralized index servers to keep indices of shared files of all peers. KaZaA utilizes cooperative super peers, each of which is an index server of a subset of peers. Some systems distribute the function of keeping indices to all peers [11].

The third search strategy is overlay topology optimization, which inspires the work we are presenting in this paper. End system multicast, Narada, proposed in [7], constructs shortest-path-spanning trees on top of a rich connected graph. Each tree rooted at the corresponding source employs the well-known DVMRP routing algorithm. Narada has proven to be a sound overlay system when the number of participants is not significant. However, because its system overheads are exponential to the size of the multicast group, it is not suitable for the P2P system, which is normally very dynamic and involves a good many nodes crossing a wide area of networks. Recently, researchers in [17] have proposed to measure the latency between each peer to multiple stable Internet servers called “landmarks”. The measured latency can then be served to determine the distance between peers. This measurement is conducted in a

global P2P domain. In contrast, we choose a completely distributed approach where distance measurement is managed in many small regions. As a result, our schemes can significantly reduce the network traffic while retaining high accuracy.

3 Unnecessary Traffic and Topology Mismatch

In a P2P system, all participating peers form a P2P network over a physical network. Maintaining and searching operations of a Gnutella peer are described in [3]. When joining a P2P network, a new peer-node gets the IP addresses of a list of existing peers from a bootstrapping node. It then attempts to connect itself to these peers as their neighbors. Once the new peer gets connected with a P2P network, it will periodically *ping* the network connections to obtain the IP addresses of some other peers in the network. Unfortunately, the join mechanism specified in a P2P network, the dynamics of peer memberships, and the nature of flooding would end up with a mismatched overlay network structure and thus incur a large amount of unnecessary traffic [12].

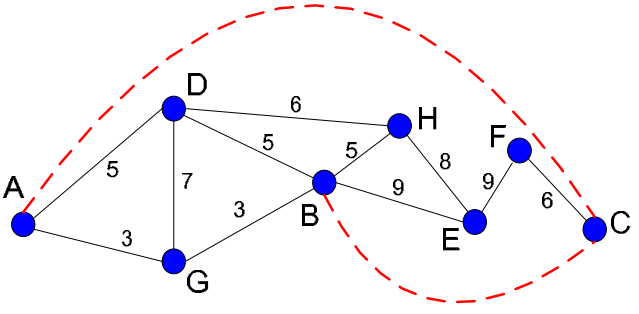

Fig. 1. An example of topology mismatch problem

An example of topology mismatch is illustrated in Fig. 1, where solid lines represent the underlying physical connections and dotted lines denote the overlay connections in a Gnutella-like P2P system. For a query message sent along the overlay path A→C→B, node B is visited twice. Although B is a peering node, B is first visited as a non-peering node when A tries to reach C. Because of the mismatch problem, the same message may traverse the same physical links, such as BE, EF and FC in Fig. 1, multiple times, causing a large amount of unnecessary traffic and increasing the P2P users' query search latency as well.

To quantitatively evaluate how serious the topology mismatch problem is in Gnutella-like networks, we simulate 1,000,000 queries on different Gnutella-like topologies with average number of neighbors being 4, 6, 8 and 10. In this simulation, we track the response of each query message to check if the response comes back along a mismatched path. We count a path as a mismatched path if a peering node on the path has been visited more than once. Result shows more than 70% of the paths are suffered from the topology mismatch problem.

We also have the following observations from the simulation. First, a query may be flooded to multiple paths that are merged to the same peer. Second, two neighboring peers may forward the same query message to each other before they receive it from the other one. In both cases, redundant query messages are generated even among logical links.

Existing studies on overlay optimization connect physically closer nodes as overlay neighbors using different techniques. However, these kinds of approaches may destroy the connectivity of the overlay and thus create many isolated islands in the P2P system. Therefore they are not feasible in unstructured P2P systems.

4 LTM and SBO

Optimizing inefficient overlay topologies can fundamentally improve P2P search efficiency. In this section, we present our solutions, LTM and SBO.

4.1 LTM

If the system can detect and disconnect the low productive logical connections and switch the connection of AC to AB as shown in Fig. 1, the total network traffic could be significantly reduced without shrinking the search scope of queries. This is the basic principle of our proposed location-aware topology matching technique[8]. Location-aware topology matching consists of three operations: TTL2 detector flooding, low productive connection cutting, and source peer probing.

Based on Gnutella 0.6 P2P protocol, we design a new message type called *TTL2-detector*. In addition to the Gnutella's unified 23-byte header for all message types, a TTL2-detector message has a message body in two formats. The short format is used in the source peer, which contains the source peer's IP address and the timestamp to flood the detector. The long format is used in a one-hop peer that is a direct neighbor of the source peer, which includes four fields: *Source IP Address, Source Timestamp, TTL1 IP Address, TTL1 Timestamp*. The first two fields contain the source IP address and the source timestamp obtained from the source peer. The last two fields are the IP address of the source peer's direct neighbor who forwards the detector and the timestamp when forward it. In the message header, the initial TTL value is 2. The payload type of the detector can be defined as 0x82.

Each peer floods a TTL2-detector periodically. We use d(i, S, v) to denote the TTL2-detector who has the message ID of i with TTL value of v and is initiated by S. We use N(S) to denote the set of direct logical neighbors of S, and use $N^2(S)$ to denote the set of peers being two hops away from S. A TTL2-detector can only reach peers in N(S) and $N^2(S)$. We use network delay between two nodes as a metric for measuring the cost between nodes. The clocks in all peers can be synchronized by

current techniques in an acceptable accuracy[1]. By using the TTL2-detector message, a peer can compute the cost of the paths to a source peer, and optimizes the topology by conducting low production cutting and source peer probing operations.

4.2 SBO

Instead of flooding queries to all neighbors, SBO employs an efficient strategy to select query forwarding path and logical neighbors [9]. The topology construction and optimization of SBO consist of four phases: bootstrapping a new peer, neighbor distance probing and reporting, forwarding connections computing, and direct neighbor replacement.

Phase 1: bootstrapping a new peer. When a new peer is joining the P2P system, it will randomly take an initial color: red or white. A peer should keep its color until it leaves, and again randomly select a color when it rejoins the system. Thus, each peer has a color associated with it, and all peers are separated into two groups, red and white. In SBO, a bootstrap host will provide the joining peer a list of active peers with color information. The joining peer then tries to create connections to the different color peers in the list. In such a way, all the peers form a bipartite overlay, in which a red peer will only have white peers as its direct neighbors, and vice versa.

Phase 2: neighbor distance probing and reporting by white peers. We use network delay between two peers as a metric for measuring the traffic cost between peers. We modify the Limewire implementation of Gnutella 0.6 P2P protocol [3] by adding one routing message type for a peer to probe the link cost to its neighbors. Each white peer broadcast this message only to its immediate logical neighbors, forms a neighbor cost table, and sends this table to all its red neighbors.

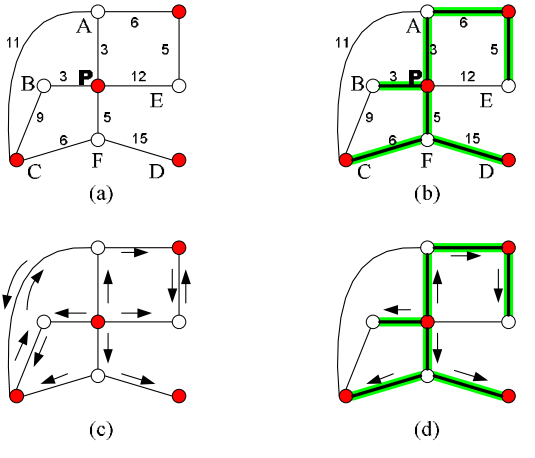

Fig. 2. An example of SBO operations

[1] Current implementation of NTP version 4.1.1 in public domain can reach the synchronization accuracy down to 7.5 milliseconds [5]. Another approach is to use distance to measure the communication cost, such as the number of hops weighted by individual channel bandwidth.

Phase 3: forwarding connections computing by red peers. Based on the obtained neighbor cost tables, a minimum spanning tree (MST) can be built by each red peer, such as P in fig. 2-(b). Since a red peer builds a MST in a two-hop diameter, a white peer does not need to build a MST. The thick lines in the MST are selected as forwarding connections (FC), while the thin lines are non-forwarding connections (NFC). Queries are forwarded only along the FCs.

Phase 4: direct neighbor replacement by white peers. After phase 3 where a MST within two hops distance is constructed, a red peer *P* is able to send its queries to all the peers within this range. Some white peers become non-forwarding neighbors, such as *E* in Fig. 2. In this case, for peer *E*, *P* is no longer its neighbor. In the phase of direct neighbor replacement, a non-forwarding neighbor, *E*, will try to find another red peer being two hops away from *P* to replace *P* as its new neighbor.

5 Performance Evaluation

To evaluate the effectiveness of LTM and SBO, we generate both physical network topologies and logical topologies in our simulation. The physical topology should represent the real topology with Internet characteristics. The logical topology represents the overlay P2P topology built on top of the physical topology. All P2P nodes are in a subset of nodes in the physical topology.

In our first simulation, we study the effectiveness of LTM and SBO in a static P2P environment where the 8,000 peers do not join and leave the system. Figures 3 and 4 show the traffic cost reduction of LTM and SBO, respectively. In these figures, the curve of 'c_n-*neigh*' shows the average traffic cost caused by a query to cover the whole network and the average number of logical neighbors is denoted as c_n. We can see that the traffic cost decreases when LTM and SBO are conducted multiple times. They both reach a threshold after several steps of optimization. LTM may reduce traffic cost by around 80-85% while SBO reduces traffic cost between 85% and 90%. However, LTM converges in around 2-3 steps while SBO needs 4-5 steps. The simulation results in Fig. 5 and Fig. 6 show that LTM reduces response time by more than 60% in 3 steps but SBO needs 8 steps to reduce 60% of the response time in a static environment.

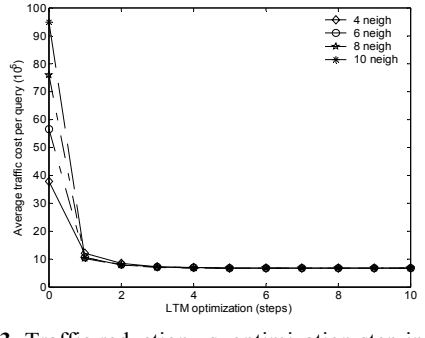

Fig. 3. Traffic reduction vs. optimization step in LTM

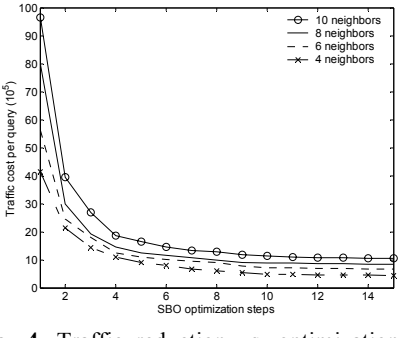

Fig. 4. Traffic reduction vs. optimization step in SBO

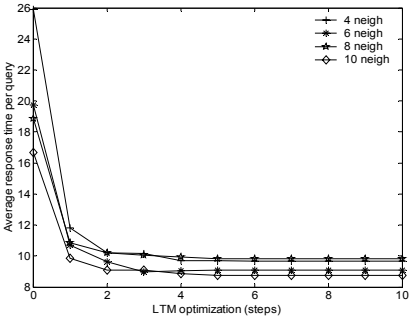

Fig. 5. Average Response time vs. opt. step in LTM

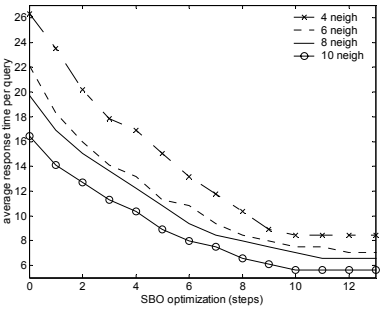

Fig. 6. Average Response time vs. opt. step in SBO

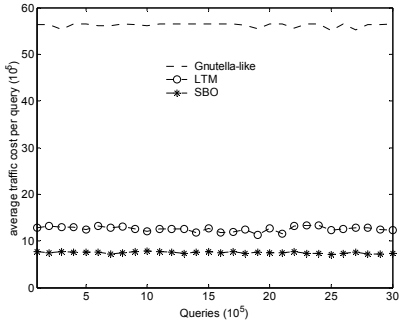

Fig. 7. Average traffic cost comparison of LTM and SBO in a dynamic P2P environment

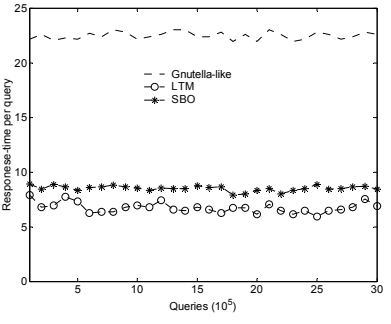

Fig. 8. Average response time comparison of LTM and SBO in a dynamic P2P environment

P2P networks are highly dynamic with peers joining and leaving frequently. The observations in [15] have shown that over 20% of the logical connections in a P2P last 1 minute or less, and around 60% of the IP addresses keep active in FastTrack for no more than 10 minutes each time after they join the system. We further evaluate the effectiveness of LTM and SBO in dynamic P2P systems. In this simulation, we assume that peer average lifetime in a P2P system is 10 minutes; 0.3 queries are issued by each peer per minute. Fig. 7 shows the average traffic cost per query of Gnutella-like P2P systems, LTM enabled Gnutella and SBO enabled Gnutella. Here the traffic cost includes all the overhead needed in the optimization steps. SBO and LTM drop the average cost by 85% and 80%, respectively. Fig. 8 plots the average query response time of each system. With the help of our carefully designed the optimization algorithms, the LTM reduces the response time to 30% and SBO decrease the response time to 35%.

6 Conclusion

We have evaluated our proposed LTM and SBO overlay topology match algorithms in static as well as dynamic environments. Both schemes are fully distributed and scalable in that each peer can conduct the algorithm independently without requesting any global knowledge. The other strength of LTM and SBO is that they are complementary to cache-based and forwarding-based approaches so that further improvements can be made when deployed together. LTM shows its advantages in convergent speed but slightly creates more overhead than SBO. It also demands synchronized time among peers, which implies that an additional overhead is needed to run a clock synchronization protocol, such as NTP.

References

[1] Fasttrack, http://www.fasttrack.nu
[2] Gnutella, http://gnutella.wego.com/
[3] The Gnutella protocol specification 0.6, http://rfc-gnutella.sourceforge.net
[4] KaZaA, http://www.kazaa.com
[5] NTP: The Network Time Protocol, http://www.ntp.org/
[6] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-like P2P Systems Scalable," *Proceedings of ACM SIGCOMM*, 2003.
[7] Y. Chu, S. G. Rao, and H. Zhang, "A Case for End System Multicast," *Proceedings of ACM SIGMETRICS*, 2000.
[8] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang, "Location-Aware Topology Matching in Unstructured P2P Systems," *Proceedings of IEEE INFOCOM*, 2004.
[9] Y. Liu, L. Xiao, and L. M. Ni, "Building a Scalable Bipartite P2P Overlay Network," *Proceedings of 18th International Parallel and Distributed Processing Symposium (IPDPS)*, 2004.
[10] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "A Distributed Approach to Solving Overlay Mismatch Problem," *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
[11] D. A. Menasce and L. Kanchanapalli, "Probabilistic Scalable P2P Resource Location Services," *ACM SIGMETRICS Performance Evaluation Review*, vol. 30, pp. 48-58, 2002.
[12] M. Ripeanu, A. Iamnitchi, and I. Foster, "Mapping the Gnutella Network," *IEEE Internet Computing*, 2002.
[13] Ritter, Why Gnutella Can't Scale. No, Really, http://www.tch.org/gnutella.html
[14] S. Saroiu, K. P.Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An Analysis of Internet Content Delivery Systems," *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, 2002.
[15] S. Sen and J. Wang, "Analyzing Peer-to-peer Traffic Across Large Networks," *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, 2002.
[16] C. Wang, L. Xiao, Y. Liu, and P. Zheng, "Distributed Caching and Adaptive Search in Multilayer P2P Networks," *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
[17] Z. Xu, C. Tang, and Z. Zhang, "Building Topology-aware Overlays Using Global Soft-state," *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, 2003.