Distributed Cycle Minimization Protocol (DCPM) for Peer-to-Peer Networks

K Srinivas[1], Dr. Gunamani Jena[2], A Satyamallesh[3]

[1]Department of Computer science and Engineering, B.V.C.Enggineering College, Odalarevu

[2]M. Tech. CSE, BVCEC

[3]Professor and HOD CSE

[1]kunchesrinu@gmail.com, [2]drgjena@ieee.org

Abstract: *In this paper, we describe the Distributed Cycle Minimization Protocol (DCMP), a dynamic fully decentralized protocol that significantly reduces the duplicate messages by eliminating unnecessary cycles. As queries are transmitted through the peers, DCMP identifies the problematic paths and attempts to break the cycles while maintaining the connectivity of the network. In order to preserve the fault resilience and load balancing properties of unstructured P2P systems, DCMP avoids creating a hierarchical organization. Instead, it applies cycle elimination symmetrically around some powerful peers to keep the average path length small.The overall structure is constructed fast with very low overhead. With the information collected during this process, distributed maintenance is performed efficiently even if peers quit the system without notification. The experimental results from our simulator and the prototype implementation on Planet Lab confirm that DCMP significantly improves the scalability of unstructured P2P systems without sacrificing their desirable properties. Moreover, due to its simplicity, DCMP can be easily implemented in various existing P2P systems and is orthogonal to the search algorithms*

Keywords: *P2P, DCMP*

I. Introduction

Peer-To-Peer (P2P) technology is attracting a lot of attention since it simplifies the implementation of large ad hoc distributed repositories of digital information. In a P2P system, numerous nodes are interconnected and exchange data or services directly with each other.

There are two major categories of P2P architectures:

- Hash based systems (for example, CAN [12] and Chord [4]), which assign a unique key to each file and forward queries to specific nodes based on a hash function. Although they guarantee locating content within a bounded number of hops, they require tight control of the data placement and the topology of the network.
- Broadcast-based systems (for example, Gnutella [3]), use message flooding to propagate queries. There is no specific destination hence, every neighbor peer is contacted and forwards the message to its own neighbors until the message's lifetime expires. Such systems have been successfully deployed in practice to form worldwide ad hoc networks due to their simplicity and versatility.

A peer-to-peer, commonly abbreviated to P2P, is any distributed network architecture composed of participants that make a portion of their resources (such as processing power, disk storage or network bandwidth) directly available to other network participants, without the need for central coordination instances (such as servers or stable hosts).Peers are both suppliers and consumers of resources, in contrast to the tradition client-server model where only servers supply, and clients consume. Peer-to-peer was popularized by file sharing systems like Napster. Peer-to-peer file sharing networks have inspired new structures and philosophies in other areas of human interaction.

Peer-to-peer networks are typically formed dynamically by *ad-hoc* additions of nodes. In an 'ad-hoc' network, the removal of nodes has no significant impact on the network. The distributed architecture of an application in a peer-to-peer system provides enhanced scalability and service robustness.

Peer-to-peer systems often implement an Application Layer overlay network on top of the native or physical network topology. Such overlays are used for indexing and peer discovery. Content is typically exchanged directly

over the underlying Internet Protocol (IP) network. Anonymous peer-to-peer systems are an exception, and implement extra routing layers to obscure the identity of the source or destination of queries.

A pure P2P network does not have the notion of clients or servers but only equal *peer* nodes that simultaneously function as both "clients" and "servers" to the other nodes on the network. This model of network arrangement differs from the client–server model where communication is usually to and from a central server. A typical example of a file transfer that is not P2P is an FTP server where the client and server programs are quite distinct: the clients initiate the download/uploads, and the servers react to and satisfy these requests. The P2P overlay network consists of all the participating peers as network nodes. There are links between any two nodes that know each other i.e. if a participating peer knows the location of another peer in the P2P network, then there is a directed edge from the former node to the latter in the overlay network. Based on how the nodes in the overlay network are linked to each other, P2P networks are classified as unstructured or structured.

Structured networks are the ones carefully designed. It usually consists of one or more servers, workstations and the underlying infrastructure that supports the network. If anything to change upgrade or move, it will affect the entire network and thus needs good planning beforehand. The administrator knows exactly what is going on at any time. No one can just come in and add a new server or workstation without authorization.

An unstructured network is simply one that doesn't have a fixed layout. An unstructured network on the other hand, it's more like a messy one. It has no rules, no working policies, anyone can add or remove or change whatever they want. Anyone who joins in becomes an integral part of the network, and changes its topology.

The advantages and disadvantages of a peer-to-peer network are as follows:

Advantages:

- It is easy to install.
- Configuration of computers is easy.

- Users can control their shared resources.
- The cost and operation of this network is less.
- It is ideal for small businesses having ten or fewer computers.
- It needs an operating system and a few cables to get connected.
- A full time network administrator is not required.

Disadvantages:

- A computer can be accessed anytime.
- Network security has to be applied to each computer separately.
- Backup has to be performed on each computer separately.
- No centralized server is available to manage and control the access of data.
- Users have to use separate passwords on each computer in the network.

Assume the network topology in Fig.1 and let peer D initialize a query message msg. D broadcasts msg to A, C, and E. C returns any qualifying results and propagates msg to A and B.

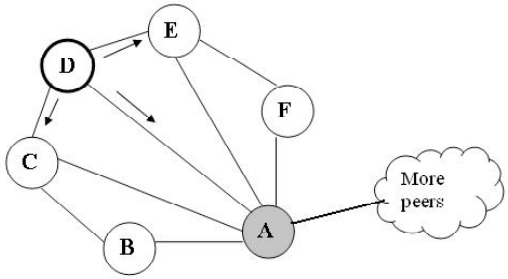

Fig.1Network with numerous duplicate messages

Similarly, E propagates msg to A and F this procedure continues until the maximum number of hops (typically seven or eight) is reached. Note that A receives the same message five times. Existing systems tag query messages with a unique identifier, and each peer maintains a list of recently received messages. When a new message arrives, the peer checks whether it has already been received through another path. If this is the case, it simply ignores the incoming message. Then the method is called Naive Duplicate Elimination (NDE).

The motivation of this work is that most real-life networks exhibit a power-law topology [8] there is a small number of peers with many neighbors (A in our example), whereas most peers have fewer neighbors. If we employ NDE in our example, most of the overhead due to duplicate elimination will occur in A. Overloading A is likely to affect many other nodes since A is the hub between the two parts of the network.

Duplicate messages severely affect the response time and scalability of P2P systems since they consume bandwidth and system resources, primarily from high-degree peers, which are crucial for the connectivity of the network. The description of Distributed Cycle minimization Protocol (DCMP), which aims at cutting the cyclic paths at strategic locations in order to avoid introducing duplicate messages in the network. In DCMP, any peer that detects a duplicate message can initiate the cutting process. This involves two steps:

- First, the peers in the cycle elect a leader, called Gate Peer.
- At the second step, the cycle is cut at a well defined point with respect to the Gate Peer.

Gate Peers are also important for maintaining the connectivity and optimal structure of the network when peers enter or quit without notification. Since any peer can become a Gate Peer via a distributed process, the system is resilient to failures.

The main characteristics of DCMP are the following:

1. It reduces duplicate messages by as much as 90 percent.
2. It requires few control messages; therefore, the overhead is minimal.
3. DCMP is suitable for dynamic networks with frequent peer arrivals and departures/failures since it is fully distributed and requires only localized changes to the network's structure.
4. There is a trade-off between eliminating the cycles and maintaining the connectivity of the network.

DCMP performs symmetric cuts and includes mechanisms to detect network fragmentation. As a result, the connectivity and average path length remain relatively unaffected. In this experiments indicate that DCMP achieves a substantial reduction in network traffic and response time, hence improving the scalability of broadcast-based P2P systems. Due to its simplicity, DCMP

can be implemented in many existing P2P systems such as. Moreover, DCMP is orthogonal to the search algorithms.

Distributed hash tables (DHTs) are a class of decentralized distributed systems that provide a lookup service similar to a hash table: (*key*, *value*) pairs are stored in the DHT, and any participating node can efficiently retrieve the value associated with a given key. Responsibility for maintaining the mapping from keys to values is distributed among the nodes, in such a way that a change in the set of participants causes a minimal amount of disruption. This allows DHTs to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures.

II. Peer-To-Peer Communication

Protocol layers may be defined in such a way that the communications within a layer is independent of the operation of the layer being being used. This is known as "peer-to-peer" communication and is an important goal of the OSI Reference Model. Each layer provides a protocol to communicate with its peer. When a packet is transmitted by a layer, a header consisting of Protocol Control Information (PCI) is added to the data to be sent. In OSI terminology, the packet data (also known as the Payload) is called a Protocol Data Unit (PDU). The packet so-formed, called a Service Data Unit (SDU) is passed via a service access point to the layer below. This is sent using the service of the next lower protocol layer.

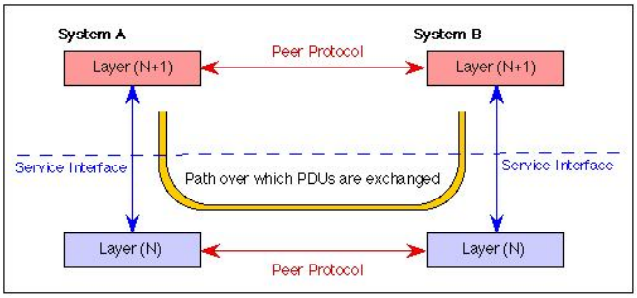

Fig. 2 Peer to peer communication using the services of a lower layer

The figure below provides an example of the OSI reference model supporting peer-to-peer communication between two End Systems (ES). In this case, the transport protocol entities communicates end-to-end using the services of the network layer below. The peer-to-peer communication takes place between the end systems using a communications protocol. In the case of the link layer, the communication takes place using the service of the physical layer. The communication takes place with the peer data link layer protocol in the next directly connected system (either an Intermediate System or an End System).

Classifications of P2P networks

P2P networks can be classified by what they can be used for:

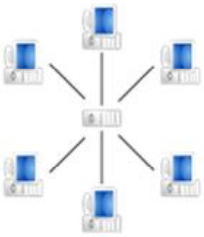

Fig 3.

File sharing
Telephony
Media streaming (audio, video)
Discussion forums

Other classification of P2P networks is according to their degree of centralization.

In 'pure' P2P networks:

Peers act as equals, merging the roles of clients and server
There is no central server managing the network
There is no central router

Some examples of pure P2P application layer networks designed for file sharing are Gnutella and Free net.

There also exist countless hybrid P2P systems:

Has a central server that keeps information on peers and responds to requests for that information.

Peers are responsible for hosting available resources (as the central server does not have them), for letting the central server know what resources they want to share, and for making its shareable resources available to peers that request it. Route terminals are used as addresses, which are referenced by a set of indices to obtain an absolute address.

e.g.

Centralized P2P network such as Napster
Decentralized P2P network such as KaZaA
Structured P2P network such as CAN
Unstructured P2P network such as Gnutella
Hybrid P2P network (Centralized and Decentralized) such as JXTA (an open source P2P protocol specification)

A. Unstructured and Structured P2P Networks

The P2P overlay network consists of all the participating peers as network nodes. There are links between any two nodes that know each other: i.e. if a participating peer knows the location of another peer in the P2P network, then there is a directed edge from the former node to the latter in the overlay network. Based on how the nodes in the overlay network are linked to each other, we can classify the P2P networks as unstructured or structured.

An unstructured P2P network is formed when the overlay links are established arbitrarily. Such networks can be easily constructed as a new peer that wants to join the network can copy existing links of another node and then form its own links over time. In an unstructured P2P network, if a peer wants to find a desired piece of data in the network, the query has to be flooded through the network to find as many peers as possible that share the data. The main disadvantage with such networks is that the queries may not always be resolved. Popular content is likely to be available at several peers and any peer searching for it is likely to find the same thing. But if a peer is looking for rare data shared by only a few other peers, then it is highly unlikely that search will be successful. Since there is no correlation between a peer and the content managed by it, there is no guarantee that flooding will find a peer that has the desired data. Flooding also causes a high amount of signaling traffic in the network and hence such networks typically have very poor search efficiency. Most of the popular P2P networks such as Gnutella and Fast Track are unstructured.

Structured P2P network employ a globally consistent protocol to ensure that any node can efficiently route a search to some peer that has the desired file, even if the file is extremely rare. Such a guarantee necessitates a more structured pattern of overlay links. By far the most common type of structured P2P network is the distributed hash table (DHT), in which a variant of consistent hashing is used to assign ownership of each file to a particular peer, in a way analogous to a traditional hash table's assignment of each key to a particular array slot. Some well known DHTs are Chord, Pastry, Tapestry, CAN, and Tulip. Not a DHT-approach but a structured P2P network is HyperCuP.

Advantages of P2P Networks

An important goal in P2P networks is that all clients provide resources, including bandwidth, storage space, and computing power. Thus, as nodes arrive and demand on the system increases, the total capacity of the system also increases. This is not true of a client-server architecture with a fixed set of servers, in which adding more clients could mean slower data transfer for all users.

The distributed nature of P2P networks also increases robustness in case of failures by replicating data over multiple peers, and — in pure P2P systems — by enabling peers to find the data without relying on a centralized index server. In the latter case, there is no single in the system.

Research in the P2P area was triggered by the success of systems like Gnutella [3] and Kazaa [5], Gnutella is a pure P2P system that performs searching by Breadth-First Traversal (BFT) of the nodes around the initiator peer. Each peer that receives a query propagates it to all of its. Neighbors up to a maximum of *d* hops. By exploring a significant part of the network, it increases the probability of satisfying the query. BFT, however, overloads the network with unnecessary messages moreover, slow peers become bottlenecks. To overcome these problems, Kazaa implements a two-layer network. The upper layer contains powerful peers, called super peers (or ultra peers) slower peers connect only to super peers. The upper layer forms a Gnutella-like network among super peers. Searching is performed by BFT at the upper layer only, since super peers contain the indices of their children. Reference [1] Contains a detailed analysis of such configurations. Yang and Garcia-Molina [2] observed that the Gnutella protocol could be modified in order to reduce the number of nodes that receive a query without compromising the quality of the results. They proposed three techniques:

- Iterative Deepening, where multiple BFTs are initiated with successively larger depths,
- Local Indices, where each node maintains an index over the data of all peers within r hops of itself, allowing each search to terminate after d " r hops, and
- Directed BFT, where queries are propagated only to a beneficial subset of the neighbors of each node.

Several heuristics for deciding these neighbors are described. This method is extended in [11] and [9], where the network is reconfigured dynamically based on the query statistics. A similar technique called Interest based Locality [6], directly contacts the most promising peer, which is not necessarily a neighbor of the query initiator. If this process does not return enough results, a normal BFT is performed. Note that none of these techniques deals with duplicate elimination, but they are orthogonal to our protocol.

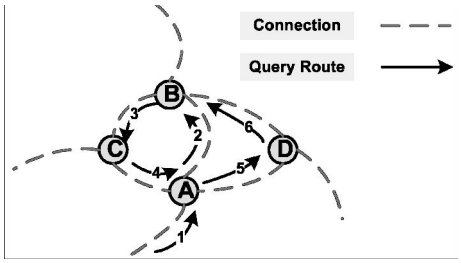

Fig 4. Example of duplicate message in Gia

Lime wire [7] maintains a table where it stores the IDs of duplicate messages and the directions (that is, neighbor peers) from where they arrive. Once a message is identified as a duplicate, it is discarded. Further message propagation avoids the directions from where duplicates have arrived. Keeping (ID, Direction) information for each duplicate message requires additional memory, especially in high degree peers as they tend to receive a lot of duplicates. Therefore, Lime wire also implements a simplified version that disables those connections from where “a lot” of duplicates are arriving. In practice, it is difficult to unambiguously define the disconnection threshold. Moreover, this method may compromise the connectivity of the Network.

Gia [14] improves the scalability of Gnutella by using a combination of topology adaptation, flow control, and 1-hop replication. Topology adaptation means that a node will prefer to connect to high-capacity peers (capacity depends on bandwidth, processing power, etc.), even by rejecting some of

its current neighbors. Gia performs search by biased random walks (RWs), where each peer forwards the query to the neighbor with the highest capacity. Nevertheless, the possibility of duplicates still exists. Consider, for instance, the network in Fig. 2.1, where the order of the peers based on capacity is A, B, C, and D (A has the highest capacity). Let peer A receive a query message. Gia routes the message as follows: A '!B '!C '!A. Therefore, A receives a duplicate.

Since A knows that it has already sent the message to B, this time, it chooses D. The message follows the path A '!D '! B thus, B also receives a duplicate. Although the message is propagated to one peer at a time, there may be many duplicates because the maximum number of hops d is much larger than that in Gnutella. Gia also implements a flow control mechanism by assigning tokens to neighbors. The aim is to prevent overloading the peers beyond their capacity. Flow control, however, allows or blocks useful and duplicate messages without distinction. Our protocol, on the other hand, can be implemented on top of Gia in order to eliminate the cycles that cause duplicates.

In order to reduce the unnecessary traffic (that is, duplicates), ACE [15] and LTM [13] use network delay as a metric to reconstruct the Gnutella network topology. In ACE, each peer uses the PRIM algorithm to build a multicast tree around itself. In LTM, each peer periodically

Broadcasts detection messages to discover and cut the connections that have the maximum delay. Disagreement of measured delay due to unsynchronized clocks causes problems when deciding the cut positions, which can influence the network connectivity. Moreover, the network delay metric mainly focuses on disabling the connections between peers that are physically far away (for example, different countries), without considering the shortcuts created by powerful peers, which are beneficial for exploiting large parts of the network.

The previous discussion applies to ad hoc dynamic P2P networks without any guarantee on the availability of resources; the majority of P2P systems in use belong to this category. Alternatively, by allowing strong control over the topology of the network and the contents of each peer, Distributed Hash Table systems (for example, CAN [1] and Chord [2]) can answer queries

within a bounded number of hops. Such configurations are outside the scope of this paper. Moreover, this paper focuses on the search process we do not consider the downloading of files after they have been located.

III. Existing System

- Simplicity, ease of deployment, and versatility
- Unstructured network topology contains many cyclic paths
- It simplifies the implementation of large ad hoc distributed repositories of digital information.
- Queries are transmitted through the cyclic path

Limitations of the Existing System

- Introduce numerous duplicate messages in the system
- It consume a large proportion of the bandwidth and other resources
- It causes bottlenecks in the entire network.

IV. Proposed System

- DCMP is dynamic fully decentralized protocol
- Queries are transmitted through the peers
- DCMP use message flooding to propagate queries
- DCMP use the method Native Duplicate Elimination (NDE)
- DCMP implements the prototype called Planet Lab

Advantage of Proposed System

Preserve the fault resilience and load balancing
It avoid duplicate messages and repeated messages
It reduce number of cycle per transaction

A. IC message

In this module, when a node generates a query message, the message is assigned a globally unique ID denoted as GUIDmsg. The main goal of our protocol is to gather information from all peers in the cycle. To achieve this, we introduce a new type of control message, called Information Collecting

(IC) message. Note that if msg travels through many cyclic paths, multiple peers will detect the duplicates. To ensure that each IC message is unique, we introduce another field, called DetectionID, which represents the direction of the connection where the duplicate was identified.

B. Cycle minimization

In this module, we choose the connection to disable in order to cut a cycle. This decision is made at the peer that receives two copies of the same IC message. This peer is the coordinator; in DCMP, any peer can act as coordinator. A straightforward way is to eliminate randomly one edge of the cycle. A node decides to cut the connection between two neighbor nodes. In order to inform the other peers in the cycle about the decision, we introduce one more message type called Cut Message (CM). CM contains the GUID and DetectionID, which are set equal to the GUID and DetectionID of the corresponding IC message. Additionally, there is a field that identifies the connection to be cut. Direction is not important in this field since any of the two nodes in the pair can disable the connection. Based on the cut message minimize or disable the path.

V. Conclusion

In this paper, we presented DCMP, a protocol for distributed cycle minimization in broadcast-based P2P systems. DCMP preserves the low diameter of Gnutella-like networks while eliminating most of the duplicate messages. The overhead due to control messages is minimal. This results in reduced response time, which, in turn, increases the scalability of the system. Our protocol is suitable for dynamic networks since it handles peer joins/departures efficiently and is resilient to failures. DCMP is also designed to be as simple as possible and is independent of the search algorithm. Therefore, it can be implemented on top of popular P2P systems such as Gnutella, Kazaa, or Gia with minimal effort. We used a simulator and a prototype implementation on Planet Lab to verify that our techniques are applicable to realistic environments.

References

[1] A. Singh, T.-W.J. Ngan, P. Druschel, and D.S. Wallach, "Eclipse Attacks on Overlay Networks: Threats and Defenses," Proc. IEEE INFOCOM '06, pp. 120-130, 2006.

[2] S. Ratnasamy, P. Francis, M. Handley, R.M. Karp, and S. Shenker, "A Scalable Content-Addressable Network," Proc. ACM SIGCOMM '01, pp. 161-172, 2001.

[3] IStoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," IEEE/ACM Trans. Networking, vol. 11, no. 1, pp. 17-32, 2003.

[4] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and Replication in Unstructured Peer-to-Peer Networks," Proc. ACM Int'l Conf. Supercomputing (ICS '02), pp. 84-95, 2002.

[5] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-Like P2P Systems Scalable," Proc. ACM SIGCOMM '03, pp. 407-418, 2003.

[6] B. Yang and H. Garcia-Molina, "Designing a Super-Peer Network," Proc. 19th Int'l Conf. Data Eng. (ICDE '03), pp. 49-60, 2003.

[7] B. Yang and H. Garcia-Molina, "Improving Search in Peer-to-Peer Networks," Proc. 22nd Int'l Conf. Distributed Computing Systems (ICDCS '05), pp. 5-14, 2002.

[8] S. Bakiras, P. Kalnis, T. Loukopoulos, and W.S. Ng, "A General Framework for Searching in Distributed Data Repositories," Proc. 17th Int'l Parallel and Distributed Processing Symp. (IPDPS '03), pp. 34-41, 2003.

[9] P. Kalnis, W.S. Ng, B.C. Ooi, D. Papadias, and K.-L. Tan, "An Adaptive P2P Network for Distributed Caching of OLAP Results," Proc. ACM SIGMOD '02, pp. 25-36, 2002.

[10] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient Content Location Using Interest-Based Locality in P2P Systems," Proc. IEEE INFOCOM '03, pp. 2166-2176, 2003.

[11] Limewire, http://www.limewire.com/, 2007.

[12] Y. Liu, Z. Zhuang, L. Xiao, and L.M. Ni, "A Distributed Approach to Solving Overlay Mismatching Problem," Proc. 24th Int'l Conf. Distributed Computing Systems (ICDCS '04), pp. 132-139, 2004.

[13] X. Liu, Y. Liu, L. Xiao, L.M. Ni, and X. Zhang, "Location Awareness in Unstructured Peer-to-Peer Systems," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 2, pp. 163-174, 2005.

[14] F. Bustamante and Y. Qiao, "Friendships That Last: Peer Lifespan and Its Role in P2P Protocols," Proc. Ninth Int'l Workshop Web Content Caching and Distribution (IWCW '04), pp. 233-246, 2004.

[15] A. Singh, T.-W.J. Ngan, P. Druschel, and D.S. Wallach, "Eclipse Attacks on Overlay Networks: Threats and Defenses," Proc. IEEE INFOCOM '06, pp. 120-130, 2006.

[16] N. Daswani and H. Garcia-Molina, "Query-Flood DoS Attacks in Gnutella," Proc. Ninth ACM Conf. Computer and Comm. Security (CCS '02), pp. 181-192, 2002.

[17] C. Gkantsidis, M. Mihail, and A. Saberi, "Random Walks in Peerto-Peer Networks," Proc. IEEE INFOCOM '04, vol. 1, pp. 120-130, 2004.

[18] S. Saroiu, P. Gummadi, and S. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," Proc. Multimedia Computing and Networking Conf. (MMCN '02), pp. 156-170, 2002.