Volume 3, Issue 5, May 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Structured Formless Probing Techniques in Peer-To-Peer Networks

Mamilla Venkatasayee *
M.Tech 2nd year, Dept of CSE,
ASCET, GUDUR, India
mvenkatasayee@gmail.com

U. Vijaya kumar
Associate Professor, Dept of CSE,
ASCET, GUDUR, India
vijay39@gmail.com

C. Rajendra
Professor & HOD ,Dept of CSE,
ASCET, GUDUR, India
hod.cse@audisankara.com

Abstract— Flooding is a fundamental building block of formless peer-to-peer (P2P) systems. In this paper, we inspect techniques to improve the performance of flooding. In particular, we present DHT P2Ps, a novel semi-organization P2P architecture with bounded peer degree. DHT P2Ps decomposes the network into different clusters, allowing peers to quickly find those neighbors which contribute much to their routing efficiency. By its link selection strategy, DHT P2Ps achieves a good performance in static and dynamic environments.

Keywords— data management, peer-to-peer networks, routing, searching

I. INTRODUCTION

While distributed hash tables (DHT) are well-studied in the research community, most peer-to-peer (P2P) systems in today's Internet are still formless. Formless architectures are attractive because of their simplicity and their high robustness. In formless systems, there is neither a centralized directory nor any control over the network topology or resource placement. When a new peer joins the P2P network, it forms connections with other peers freely, e.g., it selects arbitrary peers as neighbors. In order to publish its resources, a peer usually just stores them locally or places them on randomly chosen peers. Generally, formless overlays have loose guarantees for resource discovery, and it is possible that a file is not found although it exists in the network. It is often believed that—due to the absence of topological constraints—such formless systems have a better performance and require less maintenance overhead in highly dynamic environments where peers join and leave frequently and concurrently. Usually, these systems also support richer queries than just search by identifier, for example keyword searches with regular expressions, rangequeries, etc.

The main Achilles heel of formless P2P systems are the underdeveloped routing mechanisms. Basically, there are two fundamental routing operations:

flooding and random walks. In flooding, a search packet with a limited time to-live (TTL) maximally 10 hops in Gnutella for example is repeatedly forwarded to all neighbors, while in random walks, a packet is only forwarded to one randomly chosen neighboring peer. While a random walk is usually the less costly alternative in terms of the number of messages sent per query, the flooding approach is more robust and has better response times. This paper focuses on the flooding mechanism. However, we believe that our techniques are also useful in systems based on random walks.

The major concern about flooding is the total number of messages caused per query. More severely, in practice many of these messages are of no use and unnecessarily increase the load on the system. The main reason are redundant retransmissions: If a peer's neighbors are likely to be neighbors as well, the peer receives the same packet multiple times. In this paper, we introduce a measure to evaluate the efficiency of flooding on a given topology. We believe that this criterion captures the essence of flooding well, and also engenders many interesting theoretical questions. We then identify means to structure the topology in order to improve the quality of floods with respect to this criterion. In particular, we present DHT P2Ps, a novel semi-organization P2P network.

DHT P2Ps is a fully decentralized (local) system with undirected connections only and a limited peer degree. Beacons are used to decompose the network into clusters. Peers can orient themselves using the beacon information, and quickly find other peers which if a link to them is established significantly increase the number of peers covered by floods. If the found peer already has full degree, local link rotations are applied and also in this case the connection request can be satisfied quickly with a good neighbor. Small cycles in the topology and hence redundant messages are avoided. Moreover, **DHT P2Ps** is self-stabilizing and maintains its routing performance also in dynamic environments.

.

II. RELATED WORK

Gnutella [1] is probably the most well-known formless P2P network. However, though its success, there have been apprehension about its scalability from the beginning. Indeed, when Napster was unplugged in 2001, Gnutella broke down soon afterwards due to the inrush of former Napster users. A lot of interesting solutions have been proposed since then. In spite of the large literature about organization distributed hash tables (some authors have even proposed to implement formless data placement schemes on top of organization systems [2, 3]), many researchers have aimed at

improving the performance of formless systems, acknowledging their simplicity and their predominance in today's Internet. One active thread of research concerns content movement or replication. In particular, Cohen and Shenker [4] have shown that search is most efficient if the number of replicas is proportional to the square root of the object's popularity. The idea is that instead of (or additionally to) random neighbor connections, peers should establish connections to peers with similar interests.It has been shown that there by queries can be satisfied with a much smaller flooding radius. Many recent solutions for formless systems use alternatives for the flooding operations, for example random walks .But there have also been proposals to improve flooding itself. In the flooding is executed in several successive rounds with increasing TTL, until enough responses are received. While this solution can effectively reduce the message complexity for finding popular files, the response times are worse.Their scheme aims at minimizing the number of redundant messages by constructing a tree-like sub-overlay on which the packets are propagated. However, in contrast to our work, many connections have to be maintained which are of no use for flooding. This also implies that compared to the total number of connections in the system—the amount of peers covered by a flooding is low. More severely, the tree-like sub-overlay may result in disconnected components, especially in dynamic environments, reducing the efficiency further. In contrast, in our system, all links can be used for flooding, since they have actively been selected in consideration of their quality. Hence, while redundant messages are also rare, the flood coverage is much larger.

These systems typically assign coordinates to the different peers in order to estimate distances (in terms of latency, rather than number of hops) between a node and its (potential) neighbors. However, in this paper, we do not make use of these techniques. Finally, the idea of structuring formless systems is also used by researchers in order to avoid a mismatch of the overlay with the underlying, real network.

III.REPLICA OF P2P NETWORK

We model the P2P network as an undirected graph G = (V,E), where V is theset of peers and E the set of connections between the peers. That is, for u, v ∈ V ,{u, v} ∈ E denotes that peers u and v know the IP addresses of each other. Ther-neighborhood $\Gamma^r(v)$ of a peer v ∈ V is defined as the set of peers which are at most r hops away from peer v ∈ V , excluding v itself. For v's direct neighbors, we use the short form Γ(v) instead of $\Gamma^1(v)$. Moreover, let R be the TTL or flooding radius of the system (e.g., R ≤ 10 in Gnutella). As will be discussed inSection 4, and unlike some other formless systems, in **DHT P2Ps**, every peer has at least δ but no more than Δ neighbors, i.e., for all v ∈ V , $\delta \leq |\Gamma(v)| \leq \Delta$.

In this paper, a pure flooding algorithm is considered where each peer forwardsa packet to all its neighbors as long as the packet has a non-zero TTL. In order to maximize the probability of finding a data item or file, a flooding operationshould reach for a given radius or TTL as many peers as possible. Therefore,$|\Gamma^R(v)|$—the size of the R-neighborhood of a peer v is a natural criterion to quantify the efficiency of a flooding operation. In the following, we will refer to $|\Gamma^R(v)|$ as v's flood coverage. The flood coverage of a network network G = (V,E) is defined as the minimal flood coverage of all peers in the network. Definition 3.1. The flood coverage Ξ(G) of a topology G for a given flooding radius R is defined as

$$\Xi(G) = \min_{v \in V} |\Gamma^R(v)|.$$

Of course, not every network topology is equally suited for flooding. If a peer has neighbors which are also neighboring, many redundant messages are sent which do not increase the propagation scope. In an optimal topology G, the number of peers reached grows exponentially per hop, and if all peers have degree Δ, it holds that

$$\Xi(G) = \min\{ \sum_{i=1}^{Li=R} \Delta(\Delta-1)i - 1 \, , |v|$$

Observe that our definition of a network's flood coverage is somehow related to the important criterion of graph expansion. However, there are two crucial differences: First, we are not concerned with the expansion of all subsets of peers, but of single peers only (subsets of size one). And second, for these peers the entire R-neighborhood is considered (instead of just their immediate neighbors).

An ideal topology achieving maximal flood coverage must have a large girth, i.e., large minimal cycles. While finding such graphs is an interesting research area on its own (cf. [10] for an explicit construction), we do not follow these theoretical considerations further but go on and describe our semi-organizationP2P system DHT P2Ps which strives—in a decentralized manner—for creating topologies with large flood coverage.

IV PROBING STRICTLY STRUCTURED P2PS

In a strictly structured system, the neighbour relationship between peers and data locations is strictly defined. Searching in such systems is therefore determined by the particular network architecture. Among the strictly structured systems, some implement a distributed hash table (DHT) using different data structures. Others do not provide a DHT interface. Some DHT P2P systems have flat overlay structures; others have hierarchical overlay structures.

A DHT is a hash table whose table entries are distributed among different peers located in arbitrary locations. Each data item is hashed to a unique numeric key. Each node is also hashed to a unique ID in the same key space. Each node is responsible for a certain number of keys. This means that the responsible node stores the key and the data item with that key or a pointer to the data item with that key. Keys are mapped to their responsible nodes. The searching

algorithms support two basic operations: lookup(key) and put(key). lookup(k) is used to find the location of the node that is responsible for the key k. put(k) is used to store a data item (or a pointer to the data item) with the key k in the node responsible for k. In a distributed storage application using a DHT, a node must publish the files that are originally stored on it before these files can be retrieved by other nodes. A file is published using put(k).

In this section, searching in non-hierarchical (flat) DHT P2Ps is briefly overviewed. Then searching in hierarchical DHT P2Ps and non-DHT P2Ps are discussed in detail. More about non-hierarchical DHT P2Ps can be found in a comprehensive survey.

4.1 DHT P2Ps

In this section, we introduce the basic techniques used in **DHT P2Ps** for creating topologies with large flood coverage. As described in Section 3, a peer should connect to peers of different areas of the network, such that the shortest path between two neighbors π^1,, $\pi^{11} \in \Gamma(\pi)$ of a peer π except for the one via π itself is long.

However, in formless P2P systems, if a peer π learns about another peer π^1, π has a priori no information about π^1's location in the network, and thus also not about the hop distance between π and π^1 . Therefore, π can not decide whether it is useful to connect to π^1 , or whether its flood coverage using the existing neighbors is better.

V SIMULATION

We have analyzed the flood coverage of the topologies created by **DHT P2Ps** by simulation for up to 1 million peers. Our tests mainly focused on the parameter space $\Delta \in \{4, ...,7\}$ (each with $\delta = \Delta - 1$) and $R \in \{5, ...,10\}$. Although choosing Rd smaller than Rb gives a deterministic guarantee for the

minimal girth, this feature can not be exploited fully since it is very expensive: The amount of beacon information per peer grows quickly as longer shortest paths are enforced this way. However, such a hard guarantee seems not to be necessary, as already Rd= Rb− 2 yields very good results: Since

Rd< Rb, a peer has enough beacons in its neighborhood for orientation, but also not too many even in case of a large beacon radius Rb. Generally, Rb should roughly equal—or be slightly larger than—R, i.e., Rb≈ R.

In our simulations, the following neighbor discovery algorithm has been used:

In order to find an additional neighbor, a peer π in the **DHT P2Ps** network sendsan exploration packet of only a small constant TTL (e.g. 10 hops). This packet contains information about the beacons in the neighborhood of π, plus a section where already visited peers on the path can be stored. A peer π^1

which receives this packet forwards it to its neighbor π^{11} which—without the connection$\{\pi^1$

, $\pi^{11}\}$—shares the least beacons with π. Neighbors which are already contained in the path are avoided, and if several neighbors are equally well-suited,a random one is chosen. We have compared **DHT P2Ps** to two other strategies: a Gnutella-like strategy and a random walk strategy. In the Gnutella-like strategy, in order to find new peers it can connect to, a peer asks its neighbors for their neighbors. This procedure is repeated recursively, until the peer reaches its desired degree. In the random walk strategy, a peer sends a discovery packet with the same TTL as **DHT P2Ps**. However, unlike in **DHT P2Ps**, this packet is always forwarded to a random neighbor, and does not benefit from the beacon information for orientation.In all our simulations, the performance of the Gnutella-like strategy was ofcourse poor: The flood coverage was up to one hundred times worse than thecoverage of the other two strategies. The resulting topologies were highly clustered, and the neighbors' neighbors were often adjacent. While the random walk strategy had a much better flood coverage than the Gnutella-like strategy, it was outperformed by **DHT P2Ps** where peers quickly reached much more distant neighbors. Finally, since existing P2P systems often use longer random walks in order to find new neighbors, we have also studied a different scenario. Thereby, both **DHT P2Ps** and the random walk strategy chose neighbor candidates uniformly and at random from the entire network. Such a uniform sampling can be achieved by sufficiently long random walks. **DHT P2Ps** outperformed this random graph,albeit only by up to slightly more than 10%. (Of course, for very large graphs where only a small fraction of peers can be reached by a flooding, the difference of the coverage of the two graphs diminishes. Similarly, for very small graphs where all peers can be reached, the flood coverage is the same.) However, we believe that this uniform sampling scenario is not realistic in practice, as the mixing times and thus the length of the random walks are large and also difficult or even impossible in dynamic environments!—to compute. (Note that this computation requires a good estimation of the total number of peers in the system.) Furthermore, the probability that an exploration packet is lost is high for long walks, and it is necessary to send several redundant packets in parallel. Therefore, **DHT P2Ps** does not apply this neighbor discovery strategy but only sends packets with small TTLs, as described above. In conclusion, our first in vitro evaluation results are promising both with respect to the search efficiency and with respect to the messages' sizes. Of course, we are aware that many issues such as the dynamics of the system remain to be analyzed in detail in future work. Moreover, there is a wide variety of other alternative approaches to which have not compared **DHT P2Ps**. Although we plan to perform such comparisons, our focus here is rather on the introduction of novel ideas to improve flooding than on proposing a complete and ready-to-use system; in fact, **DHT P2Ps** can be enhanced by adding many existing heuristics, for example by introducing some form of replication, or by the extensions discussed in the next section.

VI EXTENSIONS

The basic system as described in Section 4 can be extended in several ways. In this section, we briefly discuss two possible enhancements.The first enhancement concerns the clustering. So far, there is only one level of beacons. It might be beneficial to organize the beacons in a hierarchy of several levels with increasing radius of responsibility. If a peer looks for a distant peer to connect to, it can choose the candidate with which it has only high-level beacons in common.

A small beacon hierarchy (three or four levels) might already do the job: Since floods have a small constant radius, the optimal flood coverage can also be achieved with peers which are not very far away.There are several challenges. In particular, the hierarchy must be easily maintainable when peers (and thus also beacon peers) join and leave. Moreover, it should be no disadvantage to be a high-level beacon (fairness property), and information about all beacons must be propagated efficiently. The second enhancement concerns the size of the transmitted messages. Besides general compression mechanisms, the use of Bloom filters is appealing:

Sending only the Bloom array instead of the entire beacon identifiers can reduce the burden on the system's resources (memory and bandwidth) while stillyielding acceptable probabilistic guarantees.

VII. CONCLUSIONS

This paper has embarked on identifying techniques to make flooding on formless P2P topologies much more efficient. Unlike other systems which only combat the symptoms, we strive for avoiding bad connections from the beginning. Moreover, in contrast to many organization P2P systems, **DHT P2Ps** can be started from arbitrary network topologies, i.e., from any connected graph, and develops towards better network structures in a self-stabilizing manner. Our first evaluations are promising. Moreover, many heuristics such as smart data replication are orthogonal to our approach and could be integrated to further improve search performance An interesting feature of our techniques is that they can co-exist in networks with normal clients (e.g., Gnutella clients); it is not necessary for the existing clients to know about our new clients. What is more, the entire network will benefit from our neighbor selection of—possibly a small number of—new clients, as many existing clients will experience a larger fan-out as well. We plan to investigate efficient flooding topologies further, addressing for instance **DHT P2Ps**'s dynamics: Does the system require measures in order to stabilize quickly, and if yes, which mechanisms are best? The ultimate goal is to have a running **DHT P2Ps** client which collaborates seamlessly with other formless P2P clients. Finally, we believe that our clustering approach may be interesting in other areas of distributed computing as well.

REFERENCES

1. H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems", *Communications of ACM* , Vol.46, No.2, 2003.
2. C. Yang and J. Wu, "A dominating-set-based routing in peer-to-peer networks," *Proc. of the 2nd International Workshop on Grid and Cooperative Computing Workshop (GCC'03)*, 2003.
3. N. Daswani, H. Garcia-Molina, and B. Yang, "Open problems in data-sharing peer-to-peer systems", *Proc. of the 9th International Conference on Database Theory (ICDT'03)*, 2003.
4. D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rolins, and Z. Xu, "Peer-to-peer computing", *HP Lab technical report*, HPL-2002-57, 2002.
5. D. Barkai, "Technologies for sharing and collaborating on the net", *Proc. of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, 2002.
6. B. Yang, and H. Garcia-Molina, "Improving search in peer-to-peer networks", *Proc. of the 22nd IEEE International Conference on Distributed Computing (IEEE ICDCS'02)*, 2002.
7. Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peerto -peer networks", *Proc. of the 16th ACM International Conference on Supercomputing(ACM ICS'02)*, 2002.
8. A. Crespo, and H. Garcia-Molina, "Routing indices for peer-to-peer systems", *Proc. of the 22nd International Conference on Distributed Computing (IEEE ICDCS'02)*, 2002.
9. S. C. Rhea, and J. Kubiatowicz, "Probabilistic location and routing", *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'02)*, 2002.
10. V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-yazti, "A local search mecha nism for peerto- peer networks", *Proc. of the 11th ACM Conference on Information and Knowledge Management (ACM CIKM'02)*, 2002.
11. D. Tsoumakos, and N. Roussopoulos, "Adaptive probabilistic search in peer-to-peer networks", *Proc. of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, 2003.
12. D. Tsoumakos, and N. Roussopoulos, "Adaptive probabilistic search in peer-to-peer networks", *technical report*, CS-TR-4451, 2003.
13. D. Tsoumakos, and N. Roussopoulos, "A comparison of peer-to-peer search methods", *Proc. of 2003 International Workshop on the Web and Databases*, 2003.
14. L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks", *Physical Review*, Vol. 64, 046135, 2001.

