Data Dissemination in Mobile Peer-to-Peer Networks

Thomas Repantis Vana Kalogeraki
Department of Computer Science & Engineering
University of California, Riverside
Riverside, CA 92521
{trep, vana}@cs.ucr.edu

ABSTRACT

In this paper we propose adaptive content-driven routing and data dissemination algorithms for intelligently routing search queries in a peer-to-peer network that supports mobile users. In our mechanism nodes build content synopses of their data and adaptively disseminate them to the most appropriate nodes. Based on the content synopses, a routing mechanism is being built to forward the queries to those nodes that have a high probability of providing the desired results. Our simulation results show that our approach is highly scalable and significantly improves resources usage by saving both bandwidth and processing power.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Algorithms, Design, Performance

Keywords

Mobile peer-to-peer networks, Bloom filters, content-driven routing, adaptive data dissemination.

1. INTRODUCTION

Mobile ad-hoc networks composed of mobile devices such as laptops, cellphones and PDAs with limited communication power and transmission range have emerged as a widely deployable infrastructure without the need of centralized support. The typical characteristic of these networks is that the users are interested in receiving data and services available in their vicinity, or would like to be notified about local events that are pertinent to their interests. Retrieving this information from access points stored at certain locations (*i.e.,* infostations that are geographically distributed, covered by high-speed connections) has several shortcomings: Apart from the infrastructure installation and maintenance costs, such information access points may not be available everywhere. Users often find themselves in rural or urban areas (such as tunnels or military environments) where an infostations infrastructure is not accessible. Furthermore, due to the frequent movement of the mobile nodes, data updates would have to be collected in information access points in a centralized fashion, which might result in stale data. This is particularly the case in situations where data changes dynamically, like information about traffic congestion. Such an infostations infrastructure would be more suitable for applications with loose delay constraints and high data rate requirements, in which intermittent connectivity is expected and tolerated by the users. Thus, enabling mobile devices to form unstructured ad-hoc networks, dynamically self-organize and communicate in a peer-to-peer fashion is essential for data dissemination in ad-hoc networks. Mobile nodes that are in the transmission range of each other can communicate with their peers directly. To communicate with peers outside the transmission range, messages are propagated across multiple hops in the network.

However, the transient nature of the mobile connections due to the frequent movement of the mobile nodes, as well as the heterogeneity of mobile devices pose new challenges to traditional peer-to-peer data dissemination and query propagation mechanisms. As users move, their devices may keep establishing several short-lived connections to other peers along the way and thus become bombarded by unnecessary event notifications or advertisements about locally available services and data. Even if a user is not highly mobile, the amount of forwarded queries from other mobile devices can be overwhelming, especially for devices with lower processing and communication capabilities. When disseminating data in the network, the primary goal is to reach users with the same interests while keeping the number of propagated messages small.

To address these problems, we propose *adaptive content-driven routing and data dissemination mechanisms in mobile peer-to-peer networks*. In our mechanism nodes build and maintain content summaries of their local data and adaptively disseminate them to the most appropriate peers. A peer can then use these summaries to determine if one of its peers can provide the requested data or services. Hence, peers choose to maintain summaries of other peers' content, in order to be able to efficiently locate information they need. Content summarization is recently receiving a lot of attention as a means to reduce latency, balance the query load and alleviate hot spots. By having access to content

The definitive version was published in 1-59593-041-8/05/05, MDM 2005.
http://doi.acm.org/10.1145/1071246.1071279

summaries, a node can perform a local search to determine which nodes have the requested information and thus can efficiently decide where to propagate a query, to maximize the probability for a fast reply. However, when using the content summaries, it is important to intelligently decide to which nodes and how often to propagate them to the network. Since content summaries are passed around in messages, they introduce some performance cost. Storing the summaries of the contents of all the peers in one node in the network is impossible due to bandwidth and storage limitations and also because of the dynamic behavior of the peers. In a mobile environment, changes to the stored data happen more often than they can be communicated to a single peer. Thus, the peer-to-peer network can greatly benefit from intelligent decisions regarding when and where content summaries are disseminated. Our major contributions in this paper are:

1. We propose a **content-driven routing mechanism** for finding data objects in large-scale, unstructured peer-to-peer networks. Our mechanism propagates the queries to those peers that have a high probability of providing the desired results. The mechanism is driven by content synopses that are stored locally at the peers.

2. We propose **adaptive data dissemination algorithms** that adaptively decide to which peers to propagate the content synopses to improve the search and retrieval of the objects and make more efficient use of the bandwidth and processing power resources. Our mechanism accommodates peer additions and removals from the network.

3. We present an extensive experimental study of large-scale networks, that illustrates that our mechanism reduces the number of messages sent, as well as the number of peers contacted, and achieves high recall efficiency, even in the presence of disconnecting nodes.

The remaining of this paper is organized as follows. Section 2 presents an overview of the system model and the content-driven routing and content synopses dissemination mechanisms. In Section 3 we describe the experimental evaluation of our approach and our results. Section 4 discusses related work and Section 5 concludes the paper.

2. SYSTEM DESIGN AND MECHANISMS

We consider a network of N nodes (peers) that store objects[1]. Each peer has a globally unique identifier and maintains connections with other peers. The network is unstructured, decentralized and self-organizing, meaning that peers make their own decisions to which peers to connect to or to query for objects. The number of connections of a peer can vary and is typically restricted by the resource capabilities of the peer. Each object is uniquely identified by the means of intrinsic references [5] which are generated when the object is first inserted in the system. Intrinsic references are based on the hash digest of the object's actual contents

[1]We use the term "object" to refer to data, services or events.

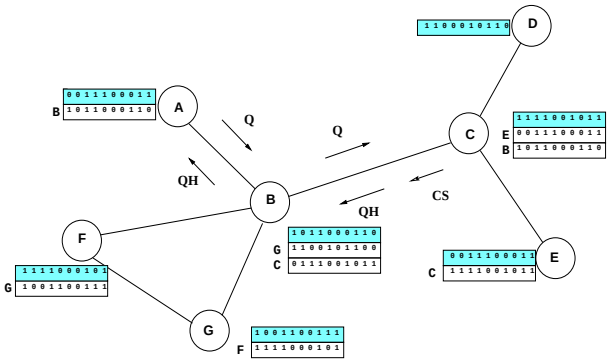

Figure 1: System operation example. Each node maintains a local content synopsis, as well as content synopses of remote peers. In this example, peer C propagated its content synopsis CS to peer B. B based on CS was able to route peer A's query Q only to C, and the result QH is routed back to A.

rather than its name or location and therefore allow us to create persistent, state-independent, and immutable storage. Alternatively, each object can be associated with a set of keywords to allow meta-data types of searching. The mechanisms presented in this paper are orthogonal to the type of search and therefore we just focus on searching by an object's intrinsic reference.

2.1 Content Synopses

Each peer uses the *Bloom filter* data structure [1] to build a synopsis of the content in its local store. Assume that peer p has a group of n objects given by the set $S_p = a_1, a_2, ..., a_n$. The Bloom filter that represents the set S_p is described by a bit array BF_p of length m, with all bits initially set to 0. We assume k hash functions, $h_1, h_2, ..., h_k$ with $h_i : X \rightarrow 1...m$. Each hash function maps each element of the set S to a value between $1...m$ in a totally random fashion. For each element $s \in S$, the bits at position $h_1(s), h_2(s), ..., h_k(s)$ are set to 1. To determine whether a certain element x is in S, we check whether all the bits given by $h_1(x), h_2(x), ..., h_k(x)$ are set to 1. If any of them is 0, then we are certain that the data item x is not in the set S. If all $h_1(x), h_2(x), ..., h_k(x)$ are set to 1, we conclude that x is in S, although there is a certain probability that we are wrong. This is the case that a Bloom filter may yield a *false positive*. Our system exploits the probability that a small number of false positives does not greatly affect the performance of our searching mechanism. This fact makes the Bloom filter approach highly suitable for locating objects accurately and fast. Our system's operation is illustrated in Figure 1.

To support the removal of members from the sets represented by the Bloom filters we use counting Bloom filters. In this approach, a counter is added to each bit in the filter, so that the number of objects that are hashed in the same position is counted. An example of a counting Bloom filter is shown in figure 2 (i).

Each peer stores two types of filters, a *local filter* for the objects available locally at the node and *remote filters* for objects stored in remote peers, indexed by their IDs. Hence,

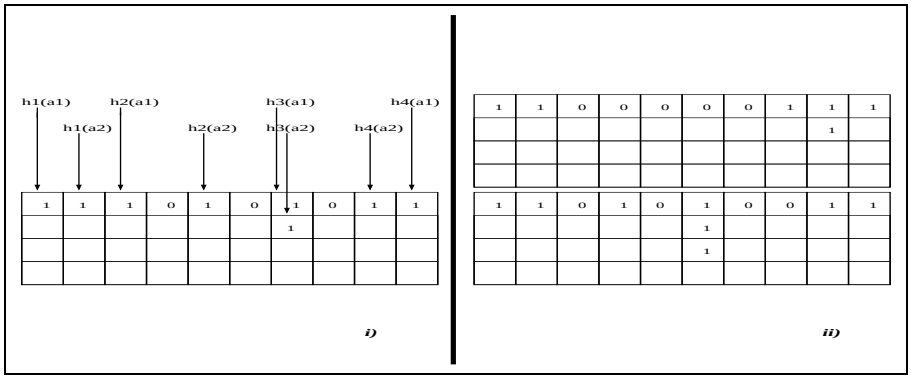

Figure 2: (i) Counting Bloom filter: The counters keep track of the number of objects that are hashed in the same position. (ii) Multi-level Bloom filter: The filter of each level is appended to that of the previous level.

to store multiple content synopses, we use multi-level Bloom filters. Figure 2 (ii) shows an example of a multi-level Bloom filter. Notice that the Bloom filter of each level is not merged but appended to that of the previous level. That approach consumes more memory space to store the Bloom filters, but allows us to estimate the location of a larger number of objects more accurately.

2.2 Content-Driven Routing

In our content-driven query routing mechanism each peer stores the content synopses of other peers, and utilizes that information in order to route queries more efficiently. In particular, when a peer receives a query, apart from searching its local content, it also searches the stored content synopses of other peers. If there is no match in its local content, the peer forwards the query only to its immediate peers whose synopses indicate that they or their neighbors contain the requested object. Only if the object is not found in any content synopsis, is the query forwarded to a set of random neighbors.

If the query cannot be satisfied locally, the node must decide to which of the peers to propagate it next. Thus, it searches the contents of the stored synopses of remote peers. The results are ranked based on the goodness of the comparison and the query message is propagated only to the peers with the highest ranks. These are the peers whose synopses indicate that they contain the requested object. If the object is not found in any content synopsis, the node forwards the query to a random subset of the immediate peers. To provide a termination condition so that messages are not propagated indefinitely in the network when no objects are found, each message is associated with a time_to_live (TTL) field that represents the maximum number of times the message can be propagated in the network. The TTL value is decreased each time the message reaches a peer. A node that receives a message with TTL zero, stops forwarding the message. Also, if a node receives the same message from two different peers, it detects the duplicates and discards the second message.

During the system operation, the node keeps statistics about the queries and the replies generated or propagated through the peer. In particular, it keeps track of (1) the number of queries sent by the peer and the replies (query hits) received to its queries from other peers, (2) the number of queries received at the peer and the replies it generates to other peers. These are used to decide to which peers to disseminate a synopsis of the local content of the peer.

2.3 Content Synopses Dissemination Strategies

Since the bandwidth used for transferring content synopses is limited, as well as the space in nodes to store them, each peer selects only some of the other peers to disseminate its content synopsis. We have implemented and compared three different strategies for content synopses dissemination.

Disseminate local content synopsis to immediate peers (Immediate Local − IL). According to this strategy each peer sends its local content synopsis to all its immediate peers and routes queries by taking into account only the content synopses of its immediate peers. This strategy is simple, but of limited use: Since only a small number of content synopses is examined for the routing decision to be taken, a lot of the queries cannot be directed using the content synopses. The protocol then resorts to randomly choosing peers to further forward the query and thus generates a lot of traffic.

Disseminate local content synopsis to peers selected adaptively (Adaptive Local − AL). Using this more elaborate strategy, each peer sends its local content synopsis to a selection of peers, according to several parameters. Again the routing is done following the synopses of the local content of other peers. The recipients of the content synopsis of a peer are selected not only among its immediate neighbors, but also among remote peers whose queries have been answered successfully from local content in the past. The adaptive selection of the synopses recipients aims to make the content synopses available to the peers that have a high probability of using them again in the future and yet keep the number of synopses transfers limited. The parameters used to decide to which peers to disseminate the content synopses are described in a following section. As the number of content synopses used in routing is limited, this strategy is also often obliged to resort to randomly forwarding queries.

Disseminate both local and remote content synopses to peers selected adaptively (Adaptive Local Remote − ALR). This multi-level strategy differs from the

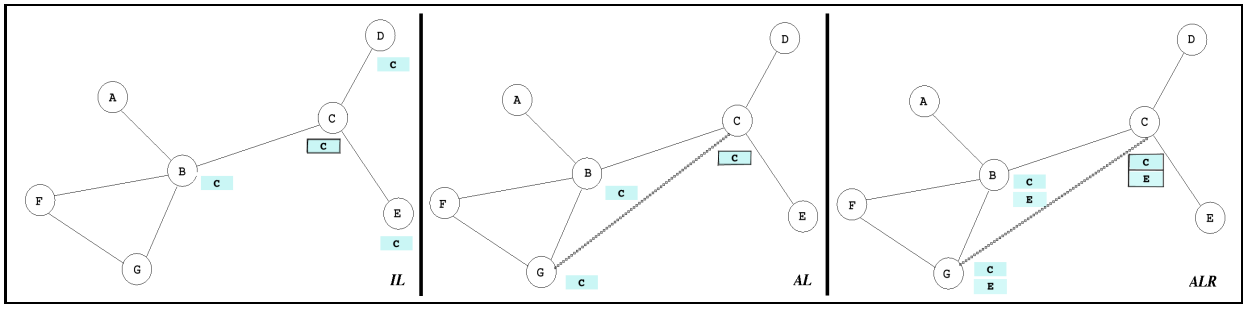

Figure 3: The content synopses dissemination strategies IL, AL, and ALR. In IL, node C propagates only its local synopsis to all peers one hop away. In AL, C propagates its local synopsis to selected immediate and remote peers. In ALR, C propagates both its local and stored remote synopses to selected immediate and remote peers.

previous, in that the peers disseminate and use for their routing decisions not only the synopses of the local content of their immediate peers or peers they have interacted with, but also synopses of the content of remote peers. More specifically when a peer propagates its local content synopsis to other peers, it also propagates the content synopses of remote peers it has stored. Other peers store those remote content synopses together with the local synopsis of that peer and use them to route queries to it. Since each peer stores and disseminates remote content synopses of peers it is connected to, it can easily route queries for content stored in them. Obviously this strategy enables the peers to examine a lot of content synopses before routing a query. Therefore a lot of the queries can be routed accurately and randomly forwarding queries is not used that often. The processing time spent in examining the content synopses is still small. The amount of information transferred between the nodes in order to disseminate the remote and the local synopses is higher than in the previous strategies, but is still restricted through the use of adaptive selection of the synopses recipients. The parameters used to decide to which peers to disseminate the content synopses are described in the next section.

Figure 3 presents an example of the different content synopses dissemination strategies discussed above. As already mentioned, all of the above strategies are assuming a simple network infrastructure, where peers route queries through their immediate neighbors. In AL and ALR a more advanced overlay network is built, where peers can open direct connections to peers that provide them with good results ("share similar interests with them") and routing can also be based on content synopses of peers outside a node's current horizon. In that case, where interest locality among the peers is exploited, queries can be routed even faster and more accurately, at the cost of managing many –probably short-lived– connections and of storing, processing and disseminating a large number of content synopses between many different peers.

Adaptive Synopses Dissemination Parameters. The AL and ALR dissemination strategies take into account the following parameters to disseminate the content synopses more efficiently.

- The number of queries q_i a node has received by a peer, and their frequency. Peers that have sent a lot of queries to that node will most probably make good use of its content synopsis in their routing decisions. A lot of forwarded queries indicate peers that route a lot of traffic. They can use a peer's content synopsis to avoid sending queries to that node for content it does not have.
- The number of replies r_i a node is providing a peer with, and their frequency. This parameter identifies the popularity of the stored objects among specific peers. Peers that generated a lot of local hits and got a lot of replies by the node to their requests will also most probably need its content synopsis in their routing decisions.

2.4 Implications of Dynamic Behavior

Since the network is dynamic and self-organizing, nodes may leave or join independently. The system must be able to disseminate content synopses to reflect such changes in the connections. Moreover content synopses must be updated whenever an object is added, deleted, or its contents have changed. In our system, updated content synopses are generated in two cases:

- When a peer detects an update at the local repository (content changes) of objects (new objects are obtained, existing objects are deleted or new versions of existing objects are created).
- When a peer detects an incoming or withdrawn peer connection (connection establishment or drop).

Content Changes. When the content is updated, a new content synopsis is disseminated by the peer. To minimize the traffic in the network our approach (1) does not generate an update unless the contents of the peers have changed and (2) groups individual Bloom filter updates into group updates to propagate them to the peers. Content synopses are disseminated due to both local and remote content changes.

Connection establishment. As nodes in a mobile peer-to-peer network move, the peers in their vicinity change.

Therefore connections are dropped and connections with new neighbors are established. We distinguish between two different modes in a peer's operation: i) When a peer is *static*, its position does hardly change and neither do the connections with its neighbors. ii) When a peer is *dynamic*, it moves a lot and its neighbors and the connections with them change constantly.

Obviously a peer can alternate between static and dynamic mode. A peer is considered static by its neighbors, when it has been connected with its peers for longer than a time threshold t_s.

Note that it is important to differentiate between static and dynamic peers as we treat them differently in the synopses dissemination strategy. When a peer moves to a new location the synopses of the contents of peers in its old vicinity are not useful in routing queries anymore, as those peers are now unreachable. Therefore pushing synopses to peers as they move would not provide any benefits, while it would result in unnecessary traffic for content synopses dissemination. When a moving peer needs to search for something or route a query, it first asks for the content synopses of its current neighbors. On the other hand, a static peer can make good use of the content synopses of its vicinity and obliging it to explicitly ask for all of them would cause unnecessary traffic. Therefore peers push their content synopses to static peers. The same rules apply to a newcomer's decision to disseminate its own content synopsis.

Connection drop. When a peer permanently disconnects from the network, neither the content synopses of other peers stored in it, nor its content synopsis stored in other peers will be useful anymore. Its immediate peers will sense the disconnected peer and all the relevant content synopses will be removed after a time threshold t_r. In addition, a `DISCONNECTED` message will be sent to the non-immediate peers to remove their corresponding content synopses.

3. EXPERIMENTAL EVALUATION

To investigate the characteristics of our adaptive content-driven routing mechanism we run extensive simulations on the Neurogrid simulator [10] using the Gnutella [8] P2P communication protocol. Our implementation of the adaptive content-driven routing protocol was done in approximately 3500 lines of Java code. The parameters used in the simulation are presented in Table 1. We chose the network size to vary up to 3000 nodes, an estimate of the number of concurrently active nodes in a university campus.

In our implementation we used counting, multi-level Bloom filters. To create the hash functions used in generating the Bloom filters, similarly to [21], we took advantage of a cryptographic message digest algorithm (SHA-1 [14]) and of its property of pseudo randomness. More specifically, we used SHA-1 to hash strings of arbitrary length, representing the peers' content, to 160 bits. We then built the hash functions by dividing the SHA-1 output into smaller sets of bits. In the experiments we used 4 hash functions, and Bloom filters that were 10 bits long. These parameter values allowed us to efficiently represent a total of 2000 unique objects on the nodes, while minimizing the number of false positives.

Our average results are derived from 20 measurements and each one of those is averaged from 20 searches (total 400 searches for each experiment).

In our **first** set of experiments we compared content-driven query routing to a traditional Breadth-First Search (BFS) algorithm. Even though BFS is not directly comparable to our content-driven routing protocols, we chose to present it here to illustrate the differences and the relative gain from our adaptive dissemination schemes.

Average message transfers during a search. Figure 4 shows that content-driven routing drastically decreases the number of query messages transferred during a search. As the number of nodes increases, the number of message transfers grows dramatically in flooding-based BFS, while the content-driven routing mechanisms manage to keep the message transfers almost at a fixed level. Thus, by using the network bandwidth efficiently, content-driven routing is able to scale to thousands of nodes. ALR, by disseminating content synopses of both local and remote peers adaptively, achieves the minimum number of message transfers needed to answer a query. It is noteworthy that the decrease in query messages between ALR and BFS reaches 97%.

Average number of nodes reached during a search. Figure 5 again shows the benefits of content-driven routing in terms of bandwidth and processing power usage efficiency. All the content-driven routing techniques are able to provide query hits by contacting more than one order of magnitude less peers than BFS, which contacts a lot of peers unnecessarily. Moreover the content-driven routing strategies keep the number of reached nodes at an almost constant level, while the nodes that are reached with BFS grow linearly as the total number of nodes increases. The figure shows that the adaptive AL and ALR techniques guide queries more efficiently than the simplistic IL, in which content synopses are disseminated blindly to all immediate peers. ALR is again the most efficient and scalable technique of all, due to the adaptive use of the multi-level Bloom filters.

Average Recall Efficiency during a search. Figure 6 shows the value of the query messages that are disseminated during a search, in terms of their contribution to the discovery of possible matches. Even though the flooding of BFS is able to discover a lot of matches, the cost of query messages transferred results in its low recall efficiency. ALR again has the highest recall efficiency, followed by the other adaptive content-driven routing strategy, AL. The reason is that adaptive content synopses dissemination places the Bloom filters where they are more likely to be needed, achieving better performance than the blind IL. As the number of nodes grows, the proportion of the total matches discovered by the content-driven routing mechanisms decreases, since the queries are guided, in order to contact a small number of nodes and to produce a small number of messages.

In our **second** set of experiments we compared the different content-driven routing protocols to each other in more detail.

Content synopses hits over misses. Figure 7 shows how much the query routing actually benefits from the use

Node Parameters	Number of nodes	Varying
Network Parameters	TimeToLive of query messages	7
	Initial number of connections per node	3
	Minimum number of connections per node	3
	Maximum number of connections per node	10
	Network topology	Random
Content Parameters	Size of pool of available objects	2000
	Number of objects per node	30
	Distribution of objects over nodes	Uniform
Bloom Filter Parameters	Size of filter, in bits	10
	Number of hash functions	4
	Size of counter for each position, in bits	4
Simulation Parameter	Number of averaged measurements	20
	Number of searches per experiment run	400

Table 1: Simulation settings.

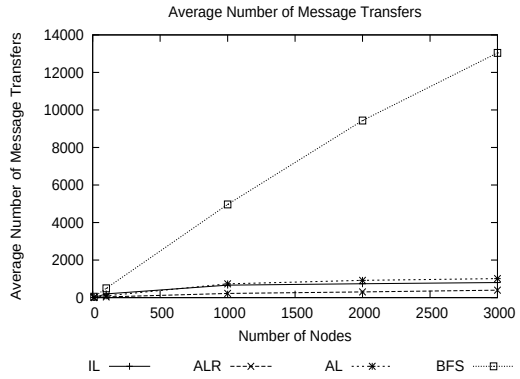

Figure 4: Average number of query messages sent during a search for different network sizes.

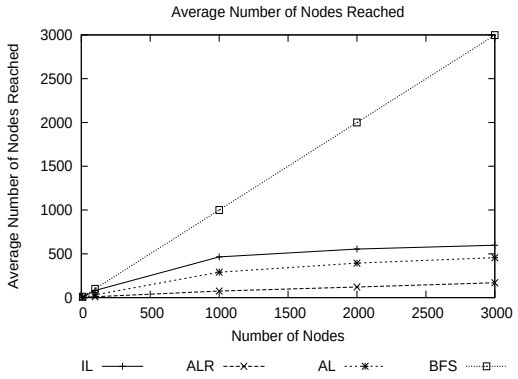

Figure 5: Average number of nodes reached during a search for different network sizes.

of the content synopses. We notice that simply placing content synopses of local content to immediate neighbors (IL) is useful for routing only about 10% of the queries. On the other hand, adaptively placing content synopses (AL and ALR) improves their usefulness to 20% for AL and to 90% for ALR. By disseminating local and remote content synopses, ALR manages to drastically decrease the number of Bloom filter misses and achieves a hits/misses ratio close to 1, meaning that half of the queries can be routed based on the content synopses.

False positives over total positives. Figure 8 shows that content-driven routing is extremely accurate. For all three routing strategies that use content synopses only a very small percentage (around 1%) of the total queries that are routed based on them is falsely routed, due to Bloom filter false positives. Thus, our choice of the Bloom filter parameters allowed us to minimize the false positives.

Total content synopses messages. Figure 9 shows the relative cost of the different content-driven routing protocols, in terms of content synopses dissemination messages. By simply disseminating content synopses only to immediate peers, IL keeps the protocol overhead low. However the usefulness of the content synopses in that approach is limited, as figure 7 indicates. AL on the other hand has to disseminate a lot of content synopses for them to be useful in query routing. ALR, by adaptively disseminating local and remote content synopses, manages to route queries effectively and yet keep the protocol overhead at a reasonable level, even as the number of nodes increases. That overhead is acceptable, if one takes into account the drastic saving of query messages ALR achieves.

In our **third** set of experiments we evaluated our protocols in a mobile environment, where peers leave the network dynamically. We gradually disconnected peers throughout the experiment run and we conducted experiments for disconnections reaching to 10, 20, and 30% of the total number of peers, which was initially 3000. We report the effects of the disconnections on the Bloom Filter behavior.

False positives over total positives. Figure 10 shows that content-driven routing remains very accurate even when a lot of peers disconnect. The neighbors of a leaving peer realize the disconnection and update their summaries, while peers further away also update their synopses when they are notified by a `DISCONNECTED` message they receive from the immediate peers. Hence false positives are not increased by the peer disconnections.

Content synopses hits over misses. Figure 11 shows that peer disconnections do not affect the success of the synopses in query routing either. ALR, which is the most aggressive mechanism in synopses dissemination, often routes queries successfully using the summaries. When a lot of

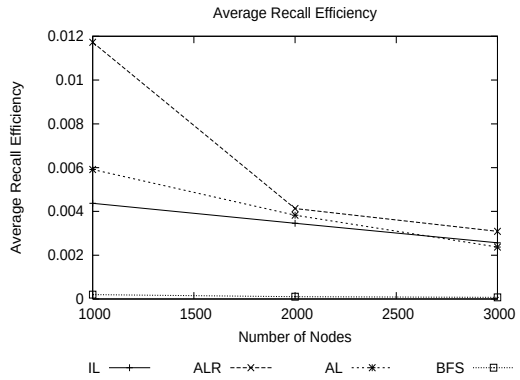

Figure 6: Average proportion of possible matches discovered over the number of queries transferred.

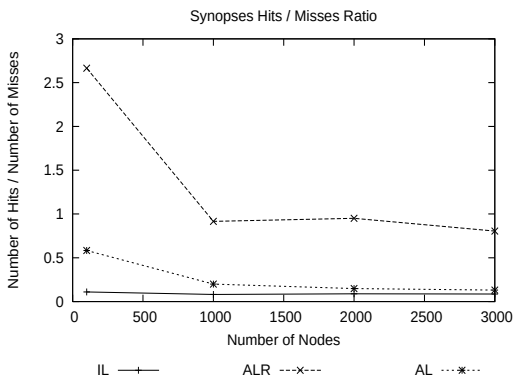

Figure 7: Ratio of hits over misses of the content synopses for different network sizes.

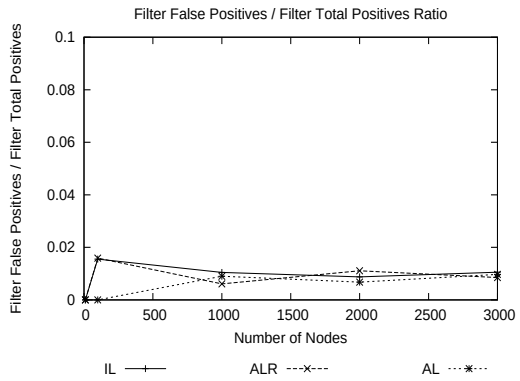

Figure 8: Ratio of Bloom filter false positives over total positives for different network sizes.

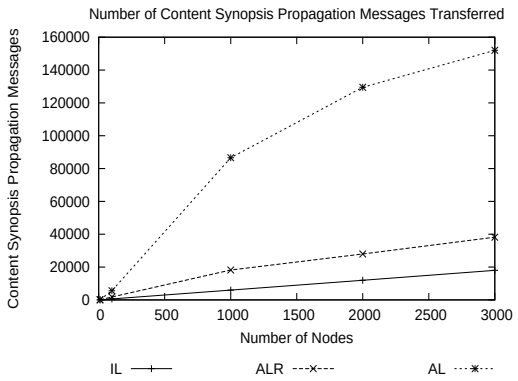

Figure 9: Total content synopses dissemination messages transferred for different network sizes.

peers disconnect, less synopses are available to help in query routing, hence a small degradation in the hit ratio.

4. RELATED WORK

Mobile ad-hoc networks emerge as an exciting paradigm of peer-to-peer collaboration and several platforms including 7DS [15] and Proem [12] have recently been proposed. Several aspects of mobile peer-to-peer operation, such as resource exchange [24] and segmented file downloading [9] have been investigated. Yet, the problem of efficiently locating objects in a mobile peer-to-peer system still remains challenging, mainly due to the unstructured nature of the network. Existing solutions from the domains of distributed databases [16] or publish/subscribe systems [2] are not directly applicable to networks of self-organizing, highly mobile nodes.

Several mechanisms have been proposed to improve searching and data dissemination in peer-to-peer networks, without however addressing any mobility issues [7, 18] For example, Planet-P [3] locates objects by replicating globally two data structures: A membership directory and a compact content (term-to-peer) index. Members gossip about changes to keep these data structures updated and loosely consistent. The cost of storing and maintaining the global data structures makes the system unsuitable for users with modem-speed connections, low storage capabilities, or for networks of more than some thousand peers. Our mechanism on the other hand does not rely on any global knowledge of the network and thus has minimum overhead and no need for structure.

Rumor spreading algorithms have been proposed, that offer probabilistic guarantees, instead of ensuring strict consistency [22]. P-Grid [4] uses a hybrid push/pull rumor dissemination algorithm. Peers that have been disconnected, that have not received updates for a long time, or that have received a pull request but are not sure if they have the latest update, pull updates from one or more other peers. In the CUP (Controlled Update Propagation) [19] protocol each node decides whether to register for receiving and propagating updates for an item according to popularity (based on the number of queries received for that item)-based incentives, either probabilistic, or log-based, also taking into account its workload and/or network connectivity.

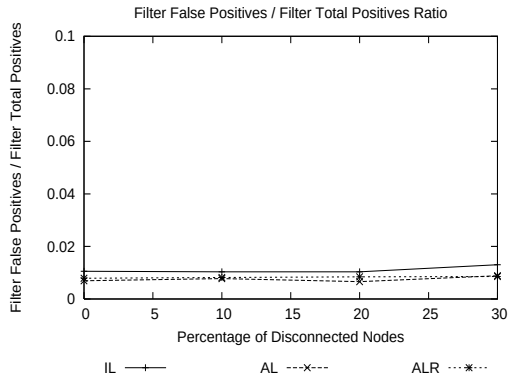

Figure 10: Bloom filter false positives over total positives for different percentages of disconnected nodes.

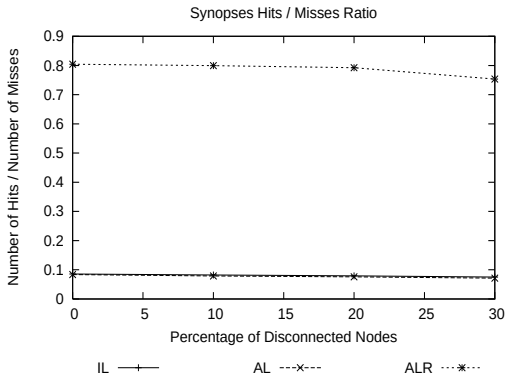

Figure 11: Content synopses hits over misses for different percentages of disconnected nodes.

There has also been work in caching the results of queries, while arbitrarily partitioning a network in layers. In [23] in addition to a local index, that keeps indices of local files, each peer maintains a response index, which caches the query results that flow through the peer. While this work also aims at reducing search traffic, the approach followed focuses on query caching and not on content summarization. Also relevant is work done on filtering and disseminating streaming data [20], where data repositories are organized hierarchically according to their coherency requirements, and work on overlay topologies for routing real-time media streams between some publishers and many subscribers. Breadth and Depth Bloom filters have also been used for summarizing hierarchical data structures [11]. These however focus on specific data structures, such as XML documents and assume a hierarchical network organization.

The Bloom Filter mechanism has also been used in Summary Cache [6] in the context of web-caching. The authors have shown that Bloom filter representations are economical and reduce the bandwidth consumption in the network.

Our work builds upon [13] and [17]. In our earlier work [13] we introduced the concept of content summarization, while in the current work we focus on mechanisms for the dissemination of the synopses, we present more elaborate summarization techniques and discuss mobility issues. One of our criteria for adaptive selection of the synopses recipients is the notion of interests, explained in [17].

5. CONCLUSIONS

In this paper we have investigated adaptive content-driven routing and data dissemination mechanisms for mobile peer-to-peer networks. Based on content synopses, nodes can forward queries intelligently only to their peers that are able to provide replies with a high probability. By disseminating the content synopses adaptively, we have shown how they can be strategically placed in the network, where they are most probably going to be needed. We have simulated mobile networks of thousands of peers and also verified the robustness of our mechanism under dynamic peer disconnections. We have compared our content-driven routing mechanism to traditional flooding-based searching to find out great savings in query messages. Thus, our approach is scalable and highly efficient in terms of bandwidth and processing power usage. We have compared three different content synopses dissemination strategies. Our results show that adaptive local and remote content synopses dissemination performs much better than blind content synopsis dissemination to immediate peers, or dissemination of just local content synopses.

6. ACKNOWLEDGMENTS

This research has been supported by NSF Award 0330481.

7. REFERENCES

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

[2] I. Burcea, H. Jacobsen, E. Lara, V. Muthusamy, and M. Petrovic. Disconnected operation in publish/subscribe middleware. In *Proceedings of the 2004 IEEE International Conference on Mobile Data Management, MDM*, January 2004.

[3] F. M. Cuenca-Acuna, C. Peery, R. P. Martin, and T. D. Nguyen. PlanetP: Using gossiping to build content addressable peer-to-peer information sharing communities. In *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing, HPDC*, June 2003.

[4] A. Datta, M. Hauswirth, and K. Aberer. Updates in highly unreliable, replicated peer-to-peer systems. In *Proceedings of the 23rd International Conference on Distributed Computing Systems, ICDCS*, May 2003.

[5] K. Eshghi. Intrinsic references in distributed systems. Technical Report HPL-2002-32, HP Labs, 2002.

[6] L. Fan, P. Cao, J. Almeida, and A. Broder. Summary cache: A scalable wide-area web cache sharing protocol. In *Proceedings of ACM SIGCOMM'98*, August 1998.

[7] L. Garces-Erice, P. Felber, E. Biersack, G.Urvoy-Keller, and K. Ross. Data indexing in peer-to-peer dht networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems, ICDCS*, March 2004.

[8] Gnutella Protocol Development. http://rfc-gnutella.sourceforge.net/, 2003.

[9] S. Goel, M. Singh, D. Xu, and B. Li. Efficient peer-to-peer data dissemination in mobile ad-hoc networks. In *Proceedings of the 2002 International Conference on Parallel Processing Workshops, ICPPW*, August 2002.

[10] S. Joseph. An extendible open source P2P simulator. *P2P Journal*, pages 1–15, November 2003.

[11] G. Koloniari and E. Pitoura. Content-based routing of path queries in peer-to-peer systems. In *Proceedings of the 9th International Conference on Extending DataBase Technology, EDBT*, March 2004.

[12] G. Kortuem, J. Schneider, D. Preuitt, T. Thompson, S. Fickas, and Z. Segall. When peer-to-peer comes face-to-face: Collaborative peer-to-peer computing in mobile ad-hoc networks. In *Proceedings of the 2001 International Conference on Peer-to-Peer Computing, P2P*, August 2001.

[13] A. Mohan and V. Kalogeraki. Speculative routing and update propagation: A kundali centric approach. In *Proceedings of the 2003 IEEE International Conference on Communications, ICC*, May 2003.

[14] National Institute of Science and Technology. Secure Hash Standard (SHA1). Federal Information Processing Standard (FIPS) 180-1, April 1995.

[15] M. Papadopouli and H. Schulzrinne. Performance of information discovery and message relaying in mobile ad hoc networks. Technical Report CUCS-004-02, Columbia University, 2002.

[16] E. Pitoura and G. Samaras. Locating objects in mobile computing. *IEEE Transactions on Knowledge and Data Engineering*, 13(4):571–592, 2001.

[17] M. K. Ramanathan, V. Kalogeraki, and J. Pruyne. Finding good peers in peer-to-peer networks. In *Proceedings of the 2002 International Parallel and Distributed Computing Symposium, IPDPS*, April 2002.

[18] S. Rhea and J. Kubiatowicz. Probabilistic location and routing. In *Proceedings of IEEE INFOCOM 2002*, June 2002.

[19] M. Roussopoulos and M. Baker. CUP: Controlled update propagation in peer-to-peer networks. In *Proceedings of the 2003 USENIX Annual Technical Conference*, June 2003.

[20] S. Shah, S. Dharmarajan, and K. Ramamritham. An efficient and resilient approach to filtering and disseminating streaming data. In *Proceedings of the 29th International Conference on Very Large Data Bases, VLDB*, September 2003.

[21] The XLattice Project. http://xlattice.sourceforge.net/, 2005.

[22] R. van Renesse, K. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21(2):164–206, 2003.

[23] C. Wang, L. Xiao, Y. Liu, and P. Zheng. Distributed caching and adaptive search in multilayer p2p networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems, ICDCS*, March 2004.

[24] B. Xu, A. Ouksel, and O. Wolfson. Opportunistic resource exchange in inter-vehicle ad hoc networks. In *Proceedings of the 2004 IEEE International Conference on Mobile Data Management, MDM*, January 2004.