CoQUOS: Lightweight Support for Continuous Queries in Unstructured Overlays

Lakshmish Ramaswamy, Jianxia Chen and Piyush Parate

Dept. of Computer Science
University of Georgia
Athens, GA 30602 USA
{laks, chen, parate}@cs.uga.edu

Abstract

The utility and the effectiveness of peer-to-peer (P2P) content distribution systems can be greatly enhanced by augmenting their ad-hoc content discovery mechanisms with two capabilities, namely a mechanism to enable the peers to register their queries and receive notifications when corresponding data-items are added to the network and a means for the peers to advertise their new content. While P2P-based publish-subscribe systems can infuse these capabilities, developing full-fledged publish-subscribe systems on top of unstructured P2P networks requires complex techniques, and it is often an overkill for many P2P applications. For these applications, we study the alternate continuous query paradigm, which is functionally similar to publish-subscribe systems, but provides best-effort notification guarantees. This paper presents CoQUOS – a scalable and lightweight middleware to support continuous queries in unstructured P2P networks. A key strength of the CoQUOS system is that it can be implemented on any unstructured overlay network. Moreover, CoQUOS preserves the simplicity and flexibility of the overlay network. Central to our design of the CoQUOS middleware is a completely decentralized scheme to register a query at different regions of the P2P network. This mechanism includes two novel components, namely cluster resilient random walk algorithm for propagating query to various regions of the network and dynamic probability-based query registration technique for ensuring that the registrations are well distributed. Our experiments show that the proposed techniques are highly effective and their overheads are low.

1-4244-0910-1/07/$20.00

1 Introduction

In recent years unstructured peer-to-peer (P2P) systems have evolved as a popular paradigm for content/resource distribution and sharing [1, 6]. Owing to the simplicity of design and flexibility towards transient node population, the real-world P2P systems are invariably unstructured. However, most unstructured P2P content distribution systems only support a very simple model for data sharing and discovery called the *ad hoc query model*. A peer that is interested in discovering data items initiates a query with a set of search parameters, which is then circulated among the peers according to the specific query forwarding mechanism employed by the network. A peer receiving a query responds to the query initiator, if it has any content satisfying the search criterion. Once a query has been processed at a node, it is removed from the local buffers (some systems *cache* recently received queries, but for a very short duration and in an ad hoc fashion). Therefore, a query exists within the P2P network only until it is propagated to various nodes and processed by them (or for a short duration thereafter, if the network employs caching). Once a query completes its circulation, the system essentially *forgets* it.

While the ad hoc query model for data discovery is essential for P2P content distribution networks, it suffers from two serious limitations. First, due to its very nature, an ad hoc query is only capable of retrieving content that exists in the P2P network during the time period when it is actively propagated and processed in the network. Further, an ad hoc query can never reach a peer that joins the network after the query has completed its circulation, and hence cannot discover matching data-items on the new peer. In this scenario, the only way for a peer to discover newly added data-items would be to repeatedly issue the same query, thereby imposing unnecessary overheads on the network. Second, the ad hoc query model provides no support for peers to adver-

tise or announce the data-items they own to other interested peers. Such capabilities are important for P2P communities where peers trade content.

These shortcomings limit the utility of the ad hoc query model for several advanced collaborative applications, such as a community of researchers sharing their recent research results or a community of amateur musicians and their patrons who are interested in buying the music produced by the musicians. In applications such as these, participating peers would not only be interested in searching for existing content, but would also want to be pro-actively informed when content matching their interests is added to the network. Further, some communities also need a mechanism through which peers can advertise their content to other interested peers. Blind broadcast of advertisement would not only result in high overheads, but could also annoy participants who would be receiving large numbers of advertisement about data-items that they are not interested in.

An approach that can partially mitigate these limitations would be to implement a *publish-subscribe (pub-sub) system* on top of the unstructured overlay network. A generic pub-sub system enables its users to register *subscriptions* expressing their interests and to announce the occurrence of certain events by *publishing* them. The pub-sub system matches incoming announcements to the existing subscriptions and notifies the users that have registered the matching subscriptions. An important point to note is that the pub-sub systems attempt to provide *guaranteed notification service* (although it might not be possible always due to system failures). We will return to this issue later in the paper.

Researchers have studied the problem of implementing P2P-based pub-sub systems on unstructured overlay networks [7, 20]. However, most of these systems require the underlying P2P networks to be organized according to specific architectures, and hence they cannot be used in generic overlays. Many of these systems also require the peers to maintain intricate index structures which add significant complexity to the design of the P2P network. This additional complexity can adversely affect the flexibility, efficiency, and scalability of the unstructured P2P system. Furthermore, it also makes the design, implementation, and management of P2P content distribution networks harder.

1.1 Contributions of the Paper

Considering the complexities involved in developing full-fledged pub-sub systems on top of generic unstructured overlays, this paper investigates the following questions: *(1) For the class of P2P applications exemplified by the content distribution scenario, do we need the notification guarantees provided by a full-fledged P2P system, or is it an overkill? In other words, does a model that provides weaker guarantees, but is much simpler and inexpensive to implement suffice for these applications? (2) If a simpler model suffices, can we design techniques and mechanisms to efficiently implement the model on top of generic unstructured overlay networks while simultaneously ensuring its effectiveness?*

It is our contention that for many applications, such as P2P content distribution, guaranteed notification is not absolutely necessary. Instead, *best effort notification*, wherein a subscription is notified of a large fraction of matching data-items and announcements, is not only sufficient but is also in tune with the design principle of P2P content sharing systems. Unstructured P2P content distribution systems like Gnutella [1] do not provide guaranteed recall. Rather, the design goal is to maximize the discovered data items, while limiting the discovery overheads. Following a similar design principle, this paper explores the *continuous* or *persistent* query model. This model provides a means through which the users can register their queries with the P2P network for extended durations of time. The P2P network provides a *best-effort* notification service for the registered queries, informing their initiating nodes of new data-items that may have been added in recent past. The continuous query model is functionally equivalent to the pub-sub model. The essential difference between the two models lies in their notification guarantees.

Towards answering the second question, this paper presents *CoQUOS* (**co**ntinous **q**ueries on **u**nstructured **o**verlay**s** – a scalable and lightweight middleware that supports continuous queries and advertisements in unstructured P2P networks. Two important features distinguish the CoQUOS system. First, it does not impose any topological constraints on the underlying P2P network and can be implemented as an independent module in any unstructured overlay network. Second, the CoQUOS system is very lightweight as it does not require complex index structures or routing techniques. Thus, it preserves the design simplicity of the unstructured overlay networks as well as their flexibility towards transient node populations – the two primary reasons for the popularity of unstructured overlays. Fundamental to our approach is a completely decentralized technique for registering and storing copies of a continuous query at various regions of the P2P network. The CoQUOS system utilizes the stored query replicas for notifying the source peers of matching advertisements that were issued by other nodes in those regions.

Specifically, this paper makes three technical contributions:

- First, we present the architectural design of the CoQUOS system for supporting continuous queries in unstructured P2P networks. In our architecture each peer maintains a set of continuous queries and notifies the respective query issuers of any matching data-items that it discovers. This architecture includes a

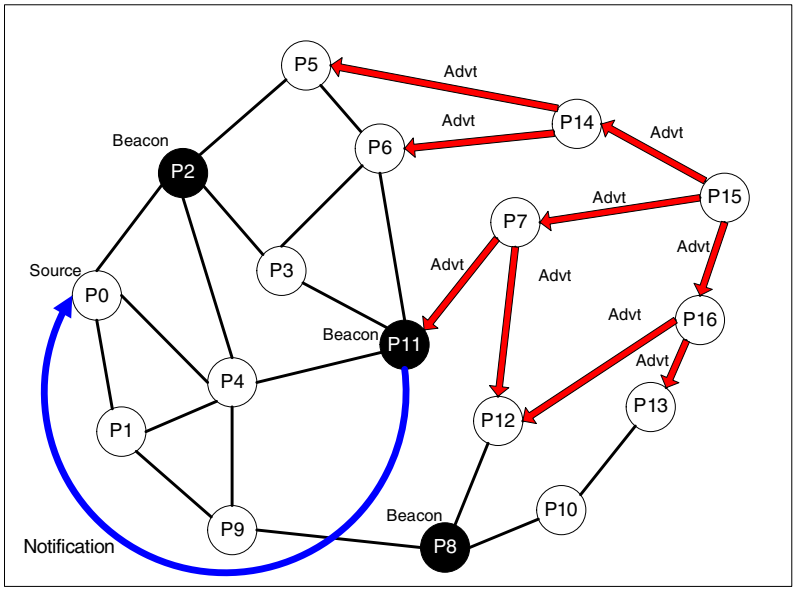

Figure 1. Overview of the CoQUOS System

completely decentralized mechanism for registering a query at various regions of the P2P network.

- Second, we propose a novel *cluster resilient random walk (CRW)* technique for propagating a query to various regions of the network. While preserving the overall framework of random walks [8], the CRW scheme favors neighbors that are more likely to send the message deeper into the network.
- Third, we design a *dynamic probability scheme* for ensuring that query registrations are well distributed along the path of the query. In this scheme, a query that has not been registered in the past several hops has a higher chance of getting registered in its next hop.

The proposed schemes are evaluated through series of experiments. The results indicate that the techniques are highly effective, and they impose very little overheads on the P2P network.

2 The CoQUOS System: Design Overview

The design goal of the CoQUOS system is to provide a highly effective notification service for continuous queries. An overlay network incorporating continuous query capabilities may independently support ad hoc queries by adopting appropriate mechanisms. However, in this paper we limit our discussion to the issue of supporting continuous queries.

Before describing the design of the CoQUOS system, we will briefly digress and discuss an important issue with respect to notification effectiveness. One way to quantify the effectiveness of notification would be to measure the overall notification success rate of the system. However, we believe that achieving high overall notification success rates is necessary, but not sufficient, in the current context. The system has to ensure reasonably high individual success rates for *all* queries. In designing the CoQUOS system, we strive to achieve this stronger notion of notification effectiveness.

The CoQUOS middleware assumes that each data-item D_r in the system has associated metadata (represented as $MData(D_r)$) that describes it, which in the current context is a list of keywords. A continuous query, represented as $Q = (SID, Predicate, VTime)$, is essentially a tuple of three components, namely the *source ID* (SID), the *query predicate* ($Predicate$) and the *validity time* ($VTime$). The source ID uniquely identifies the peer issuing the query. The query predicate is the matching condition of the query, and it is used by the source peer to specify its interests. We assume that the predicate is a list of keywords. However, the CoQUOS system can be extended to support other query predicates such as range predicates or regular expressions. Validity time ($VTime$) represents the time until which the source node is interested in receiving notifications. Peers announce their new data items through advertisements. An advertisement, represented as $Ad = (AID, MData)$ has two components. The *advertising peer ID* (AID) identifies the advertising peer and the metadata ($MData$) is the

metadata of the content being advertised.

The basic idea of the CoQUOS middleware is to maintain a continuous query at multiple locations in the network (on multiple peers). If a query Q_m is registered at a peer P_i, then P_i is called the *beacon node* of the query Q_m. A query typically has multiple beacon nodes, and analogously, a peer serves as the beacon node for multiple queries. A peer that registers a query implicitly agrees to assume the responsibility of notifying the source node of any *matching* data items that peer might discover. A data-item D_r (and equivalently its advertisement) is said to *match* a continuous query Q_m, if D_r's metadata contains *all* the keywords in Q_m's predicate. A beacon node may discover a new data item matching a query's predicate in two ways. The first is through advertisement messages it receives. In addition, the beacon node can periodically circulate the query in its close vicinity to search any new data items that might have been added. We observe that the two discovery mechanisms are functionally analogous to each other. In this paper, we assume that the beacon points discover matching data-items through advertisements.

In the current design, the advertisements are circulated through a Gnutella-like broadcast scheme. However, for efficiency purposes, the TTL of the advertisement messages are set to very low values. Advanced strategies like iterative deepening and directed breadth first search [21] can also be used for this purpose, which will further reduce the message load. Studying the effects of these strategies is a part of our future work.

Figure 1 demonstrates the functioning of CoQUOS middleware. The query issued by the node P_0 is registered at peers P_2, P_8 and P_{11}. The node P_{15} issues an advertisement that matches the query, with TTL set to 2. This advertisement reaches the beacon node P_{11}, which notifies P_0.

Our main focus in this paper is on the design of decentralized strategies for selecting good beacon node sets for the continuous queries. Factors such as churn in the overlay network and the load imbalance among the network nodes can have considerable impact on the performance of the CoQUOS system. Our technical report [13] discusses these issues and presents a set of low-cost techniques to address them.

3 Beacon Nodes Selection for Query Registration

Since the beacon nodes play a crucial role in notifying the source peer of matching data-items, the choice of beacon nodes has a significant impact on the notification success rates of a continuous query. Therefore, a key challenge is to select a set of peers to host the continuous query (i.e. become one of its beacon nodes) so that the notification effectiveness is maximized.

Towards addressing this challenge, we first examine the characteristics of a good beacon set. First and foremost, the beacon nodes of a query should be distributed in all major regions of the overlay network. This property is essential for achieving high notification success rates as the advertisements are circulated through very limited broadcast around the advertising peer. Second, to prevent duplicate notifications, the beacon nodes of a query should not be located too close to one another.

Selecting beacon nodes that satisfies the above two properties is challenging because of the highly decentralized nature of the unstructured overlay networks. The CoQUOS system includes an efficient and completely decentralized technique for beacon node selection. In this scheme, a continuous query is circulated in the network (by neighbor forwarding), and each peer that receives the continuous query decides independently whether to register and store the query.

We now address the two central questions of our approach, namely *how to propagate continuous queries so that they reach all major regions of the graph?* and; *how should a peer receiving a query decide whether to register the query?*.

3.1 Cluster Resilient Random Walks for Query Propagation

Flooding-based broadcast is a straightforward option for propagating continuous queries. However, it is a poor choice since the messages mostly remain in close vicinity of the source node and do not go deep into the network, hence failing to reach various regions of the network. A possible alternative in this regard is the *random walk* technique [8, 12]. In the context P2P networks, random walk works as follows: A peer receiving a message whose TTL has not expired forwards it to one of its neighbors completely at random. For a given message load, random walk has better ability to reach remote regions of the network when compared with flooding scheme.

The above property of the random walk makes it an attractive paradigm for propagating continuous queries. Unfortunately, the random walk protocol suffers from one significant drawback that undermines its utility for propagating queries in the CoQUOS system: Studies have shown that the performance of the random walk technique degrades considerably on networks that exhibit significant degrees of node clustering [8]. In these networks, there are distinct clusters of nodes with large numbers of connections among them. Comparatively, the links flowing across clusters are few in number. For these networks, the random walk protocol suffers from the following problem: Messages are likely to keep circulating within the clusters for large number of hops, thereby spending most of their TTL. For example, in

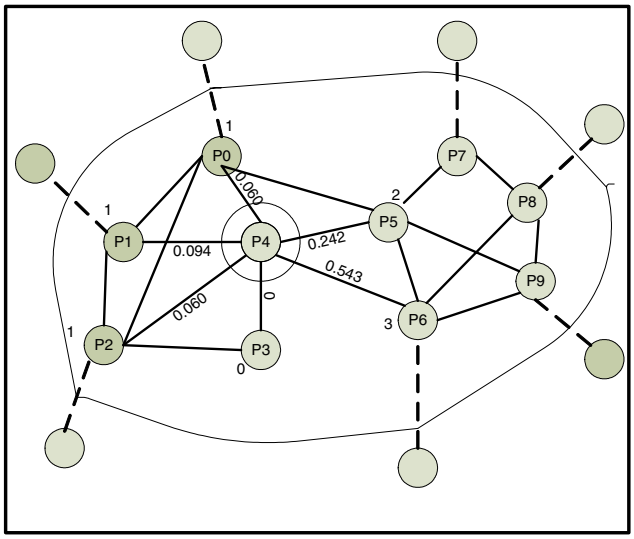

Figure 2. Illustration of Cluster Resilient Random Walk

the network shown in Figure 2, a random walk message that reaches the peer P_4 has a high probability of visiting P_0, P_1, P_2 and P_3 (peers in P_4's cluster). The cloud around the labeled nodes just indicates the part of the network we are focusing on. In other words, the message gets *trapped* in clusters, which affects its ability to reach different regions of the graph. In this paper we use the terms random walk and pure random walk (PRW) interchangeably.

To overcome the above limitation of the random walk technique, we have designed a novel message propagation scheme called the *cluster resilient random walk* (CRW) scheme. This scheme is motivated by a crucial observation: *Two peers belonging to the same cluster generally have large numbers of common neighbors*. Thus, a peer P_j has a lesser likelihood of being in the cluster of another peer P_i, if a large fraction of P_j's neighboring peers are not neighbors of P_i. Based on this observation, the CRW scheme forwards messages to *out-of-cluster* nodes with high probability, thereby mitigating the possibility of messages getting trapped in clusters. In this scheme, a peer computes the overlap between its neighbor list and those of each of its neighbors, and uses this information while making message forwarding decisions. A peer that has little overlap has higher probability of receiving the query in the next hop, and vice-versa.

Let $NbrList(P_j)$ denote the list of neighbors of P_j. Let the peers P_k, P_{k+1}, P_{k+2}, ..., P_{k+l} denote the neighbors of a node P_j. Let $UniqueNbrs(P_{k+1}, P_j)$ denote the set of neighboring peers of P_{k+1} that are *not* the neighbors of P_j (i.e., $UniqueNbrs(P_{k+1}, P_j) = NbrList(P_{k+1}) - (NbrList(P_j) \cap NbrList(P_{(k+1)}))$). Suppose the node P_j receives a query message from a neighboring peer P_k. In the CRW algorithm the probability of a neighbor P_{k+1} receiving the message in the next hop (represented as $FwdProbability(P_{K+1})$) is proportional to $(\frac{|UniqueNbrs(P_{k+1}, P_j)|}{|NbrList(P_{(k+1)})|})^{\lambda}$. λ is called the *bias factor* and it controls the extent of bias towards neighbors with larger $|UniqueNbrs|$ values. Larger values of λ induce stronger bias and vice-versa. If λ is set to 0, then there is no bias in the scheme. In this case, CRW becomes equivalent to the random walk scheme. Hence, the random walk technique can be viewed as a special case of the proposed approach. The appropriate λ value depends upon the topology of the network under consideration. In general, if the network exhibits high degree of clustering among its nodes, λ should be set to higher values (≥ 3).

Figure 2 illustrates the CRW algorithm. The cloud indicates the part of the network we are focusing on. The query message is at the node P_4 and λ is set to 2. For each neighboring node of P_4, we show its $|UniqueNbrs|$ value with respect to P_4. The numbers on the edges indicate the probabilities of forwarding the query along that link. Notice that the query has a much lower probability of remaining within P_4's cluster (comprised of $\{P_0, P_1, P_2, P_3\}$) in the next hop. In a nutshell, the query propagation in the CoQUOS system works as follows. The source peer creates a query message initializing its TTL to a default value (called query TTL). Each peer along the query's path forward the message to one of their neighbors according to their $FwdProbability$ values. The TTL is decremented at each hop, and the process terminates when the TTL becomes 0. The CRW scheme requires the peers to maintain the connectivity information of their neighbors. This necessitates additional communication between neighboring peers, since peers' connectivity evolve over time. Strategies such as *lazy updating* and *piggybacking* can considerably minimize the communication overheads [13].

3.2 Dynamic Probability-based Query Registration Scheme

The CRW scheme provides a mechanism for propagating a continuous query. But, how does a node receiving this message decide whether to register the query? The straightforward solution of registering queries at every node they visit would result in large numbers of unnecessary subscriptions, thereby affecting the efficiency of the network. Alternatively, each peer receiving a query message can decide to register it with a certain fixed probability, say Rp. We call this scheme the *fixed probability-based query registration scheme* (FP scheme, for short). Although this strategy seems intuitive, it cannot guarantee high notification success rates for every query. This is because for some continuous queries a long series of peers in the path of the query

message may all decide not to register the query, whereas another sequence of consecutive nodes may all decide to host the query. The advertisements originating near the *dry patches* of a query's path (a dry patch is sequence of nodes in a query's path with no registrations) might fail to reach any of its beacon nodes, thus leading to low success rates. Thus, there is a need for a query registration scheme that can ensure reasonable success rates for *all* queries. In other words, the success rate of even the poorest performing query should be reasonable.

Considering these requirements, we have designed a novel *dynamic probability*-based (DP) technique for peers to decide whether to register a continuous query. Unlike the fixed probability scheme, the registration probability of a query varies as the query traverses along its route. The central idea of the dynamic probability scheme can be summarized as follows: *The probability of registering a query at a peer node would be high if the query has not been registered at the nodes it visited in the recent past, and vice-versa*. Thus, the chances of having long dry patches are substantially reduced and the various registrations of query are likely to be well distributed within the overlay network.

Specifically, the scheme works as follows. Each continuous query message Q_m is associated with a value called *registration probability* ($Rp(Q_m)$). When a peer issues a query, the registration probability is set to an initial value (called *initial probability*). Each peer along Q_m's path registers it with probability equal to the current value of $Rp(Q_m)$. If a peer registers Q_m, it also resets the value of $Rp(Q_m)$ to the default initial value. Otherwise, the peer increments $Rp(Q_m)$ by a pre-determined amount (called *probability increment*). Thus, the registration probability value associated with a message keeps increasing until it gets registered at a peer, at which point it falls suddenly to the default initial value. The number of beacon nodes of a query can be controlled through the initial probability and probability increment parameters. Higher values of these parameters result in larger number of subscriptions and vice-versa.

Our experiments show that our decentralized beacon node selection scheme not only yields significant improvement in the overall success rates, but also ensures reasonably high individual success rates for all queries.

4 Experiments and Results

We have performed a range of experiments to study the performance and costs of the proposed schemes. The main goals of this experimental study are:

1. Evaluating the effectiveness of the CRW technique in reaching various regions of the network.
2. Studying the performance of the dynamic probability approach for query registrations.
3. Evaluating the communication overheads of the CoQUOS system.

Our experiments simulate the CoQUOS system on various unstructured overlay networks. For comparison purposes, our simulator also implements pure random walk and flooding-based query propagation schemes. It supports both fixed and dynamic probability schemes for query registration. For simplicity, it is assumed that each query and advertisement contain one keyword. The keywords are drawn from a corpus of $10,000$ words. The experiments reported in this paper use power-law distributions for keyword popularities, as prior studies have reported similar distributions for real-world P2P network like Gnutella [17]. The results from other distributions (like random and Gaussian) show similar trends. In our simulator, an individual peer's entry and exit are modeled as Poisson processes. λ (the bias factor for CRW) is set to 5 for all the experiments. The network topology, the query rate, the advertisement rate and the churn rate can be specified as input parameters.

	Total Nodes	Node Degree			
		Avg.	Median	Min	Max
Random	5000	10.0	10	2	24
Power-Law	5000	4.0	2	1	623

Table 1. Topological Characteristics of Networks

In this paper, we use two network topologies for our experiments: (1) A 5000 node power-law (Zipf) topology network with exponent value set to 0.9 (henceforth referred to as power-law network); and (2) A 5000 node uniform random graph (henceforth random network). We note that previous studies [16] have shown that real-world P2P networks follow power-law topologies. Table 1 indicates the key parameters of these networks. Detailed experiments involving various network topologies are available in our technical report [13].

Performance Metrics

We use two performance metrics for quantifying CoQUOS middleware's effectiveness, namely *mean notification success rate* and *minimum notification success rate*. Consider a continuous query Q_m that was issued by peer P_i. Let $MCount(Q_m)$ denote the total number of matching advertisements issued in the validity duration of Q_m. Suppose P_i was notified of $NCount(Q_m)$ of these advertisements. The notification success rate of Q_m ($NSR(Q_m)$) is defined as $NSR(Q_m) = \frac{NCount(Q_m)}{MCount(Q_m)}$. The *mean notification success rate* (mean NSR for short) of the system is defined as

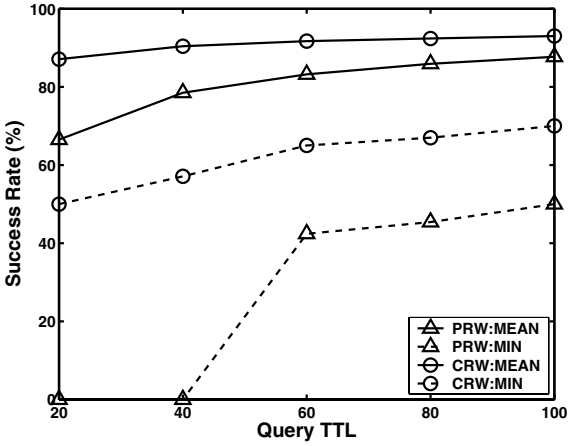

Figure 3. Comparison of NSRs of PRW and CRW (Power-law Network)

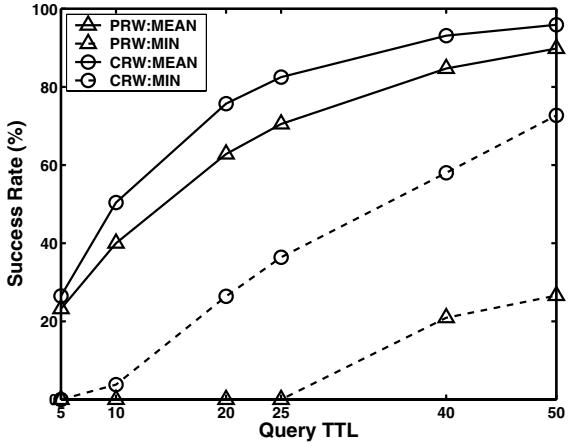

Figure 4. Comparison of NSRs of PRW and CRW (Random Network)

the average of the $NSRs$ of all the queries that were issued during the observation time duration, whereas *minimum notification success rate* (minimum NSR for short) is their minimum.

The message load in the system is measured in terms of the number of messages received by each peer in unit time. Cumulative message rate of a peer P_i is defined as the number of messages received by P_i in unit time. The cumulative message rate of the system is defined as the average of the cumulative message rates of all the peers. There are four types of messages in the CoQUOS system, namely query messages, advertisement messages, notification messages (these are messages carrying notification) and system maintenance messages (messages used for maintaining consistency of NbrLists). Accordingly, the cumulative message rate of the system is composed of four component − query message rate, advertisement message rate, notification message rate and maintenance message rates. These terms are defined similar to the cumulative message rate. Message load per query quantifies the message costs imposed by each continuous query and is calculated as the ratio of total number of query messages circulated in the system to total number of continuous queries issued.

Evaluating the CRW Algorithm

In the first set of experiments, we exclusively study the performance of the CRW algorithm for query dissemination by comparing it with PRW and flooding-based approaches. Since the goal is to study the performances of query propagation schemes, we use the fixed probability technique for query registrations for all the experiments in this series.

In the first experiment, we measure the mean and the minimum NSRs of the PRW and CRW query propagation schemes when the query TTLs are set to various values. The registration probability value of the FP query registration technique is set to 0.25. Figure 3 and Figure 4 shows the mean and the minimum NSRs of the two schemes on power-law and random networks respectively. The results show that the CRW scheme provides significant benefits both in terms of the mean and minimum NSRs. The mean NSRs of the CRW algorithm are 15% to 68% higher than the corresponding values of the PRW scheme for the power-law network. The improvements provided by the CRW scheme on the minimum NSRs are even higher. CRW yields reasonable minimum NSRs even at relatively low query TTL values. For power-law network, the minimum success rate of the CRW scheme is around 50% when the query TTL is just 20, whereas PRW achieves a similar value only when the query TTL is 100. Due to space limitations, we have not reported our experiments with Gnutella-like flooding approach for query propagation. However, flooding-based query propagation yields very low success rates, as the messages in flooding-based scheme remain in close vicinity of the source peer and do not reach different regions of the network. These results can be found in the technical report version of this paper [13].

Evaluating the Dynamic Probability Scheme

In the next set of experiments, we evaluate the dynamic probability scheme for query registration by comparing it with the fixed probability scheme. As mentioned earlier, the query registration schemes have to be used in conjunction with a query propagation scheme. In our experiment we simulate the fixed probability and the dynamic probability schemes in conjunction with both the PRW and the CRW query propagation techniques to obtain four combinations, namely *pure random walk with fixed probabil-*

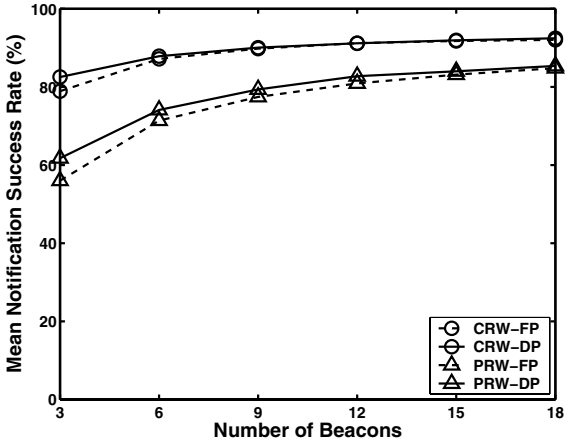

Figure 5. Comparison of FP and DP schemes on mean NSR (Power-law)

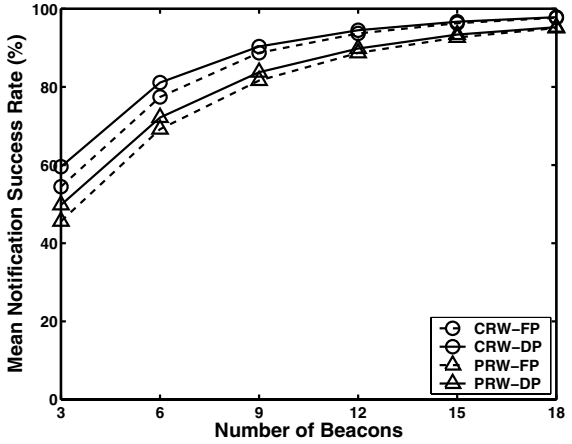

Figure 6. Comparison of FP and DP schemes on mean NSR (Random)

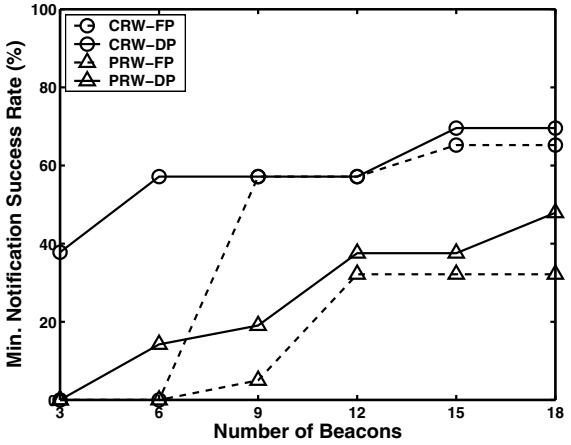

Figure 7. Comparison of FP and DP schemes on minimum NSR (Power-law)

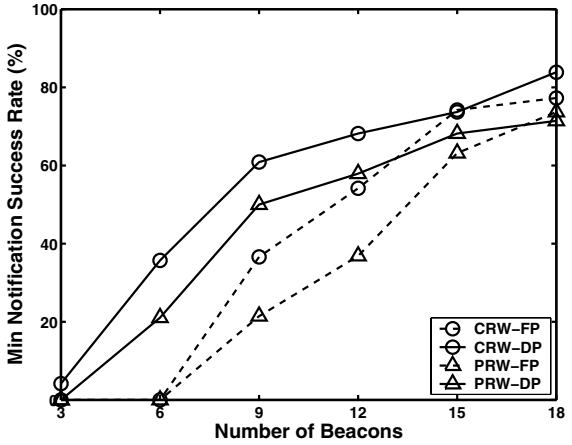

Figure 8. Comparison of FP and DP schemes on minimum NSR (Random)

ity (PRW-FP), *pure random walk with dynamic probability (PRW-DP)*, *cluster resilient random walk with fixed probability (CRW-FP)* and *cluster resilient random walk with dynamic probability (CRW-DP)*. The CoQUOS system uses the cluster resilient walk coupled with dynamic probability-based query registration scheme.

Figure 5 and Figure 6 respectively show the mean NSRs of the four combinations for the power law network and the random network. On the X-axis is the average number of beacon nodes per query (henceforth called average beacon count) and on the Y-axis is the mean NSRs. The initial probability and the probability increment parameters of the dynamic probability scheme are set to identical values. They vary from 0.007 to 0.28 for the power-law network and between 0.03 and 0.64 for the random network to yield the beacon count range. The registration probability value for the fixed probability ranges from 0.064 to 0.485 for power-law network and 0.027 to 0.73 for the random topology network. The query TTLs of the continuous queries are set to 50 for the power law network and to 25 for random network. Irrespective of the query propagation mechanism, the dynamic probability scheme outperforms the fixed probability technique for most beacon count values. The two schemes yield similar mean NSRs at higher beacon count values.

Since the differences between the mean NSRs of the two query registration schemes are relatively low, it is natural to question the usefulness of the dynamic probability scheme. However, recall that the primary motivation of the dynamic probability scheme is to ensure reasonably high success rates even for *all* queries. To evaluate whether the scheme achieves this objective, we plot the minimum NSRs of the four combinations in Figure 7 and Figure 8. The

	Query Msg. Rate	Advt. Msg. Rate	Ntfy. Msg. Rate	Mtn. Msg. Rate	Cuml. Msg. Rate
Power-law N/W	2.5	11.92	0.26	0.38	15.06
Random N/W	2.5	6.01	0.38	0.96	9.85

Table 2. Communication Costs of the CoQUOS system

results show that the dynamic probability scheme provides considerably higher minimum NSRs than its fixed probability counterpart. The improvements in the NSRs are essentially due to better distribution of beacon nodes.

Overheads of the CoQUOS System

In the final set of experiments we evaluate the communication costs of the CoQUOS system in terms of the message rates. As mentioned earlier, the cumulative message rate of the CoQUOS system is comprised of four components, namely query message rate, advertisement message rate, notification message rate and system maintenance message rate. We evaluate each of these components and study their contributions towards the cumulative message rate. We execute the CoQUOS system on the power-law and the random networks with all its capabilities enabled. The initial TTL of the continuous queries and the advertisements are set 50 and 2 respectively for both networks. The initial probability and the probability increments are both set to 0.02. The query rates and the advertisement rates of all peers are set to 0.05 per unit time and the peer churn rate is 10%.

Table 2 shows the four components of the cumulative message rate for both power-law and random networks. The advertisement message rate is the predominant component of the cumulative message rate. This implies that using advanced broadcast strategies for advertisements (such as directed BFS) can considerably reduce the message loads. The messages due to system maintenance operations form a very small fraction of the cumulative message load demonstrating the lightweight nature of the CoQUOS system.

5 Related Work

The work presented in this paper is primarily related to two fields, namely P2P networks [2, 6, 15] and publish-subscribe systems (event-delivery systems) [3, 5], both of which have been very active areas of research in the past few years.

Pub-sub systems can be classified into two broad categories: (1) topic-based – wherein users join specific topic groups in which all the messages related to the topic are broadcast; and (2) content-based – wherein users specify their interests through predicates. With the aim of enhancing scalability, efficiency and scalability several distributed pub-sub systems have been proposed [3, 5]. Recently, P2P computing models have been utilized for this purpose. Researchers have explored two strategies for constructing P2P-based pub-sub systems, namely *(a)* adopting a structured P2P network like Chord [18] or CAN [14] as the underlying substrate, and utilizing its indexing schemes for mapping subscriptions and events to nodes of the P2P systems [10, 19]; *(b)* organizing the nodes of the P2P system into specialized topologies and/or embedding application-specific distributed index structures within nodes of the P2P network [7, 20, 22]. The Sub-2-Sub system [20] organizes the peers into clusters using an epidemic-style algorithm such that nodes with similar subscriptions are put into the same cluster. The publisher of an event joins the corresponding cluster and disseminates the event to the cluster members.

The CoQUOS system differs from the above works in terms of motivation, goals and approach. The goal of the above systems is to improve the various performance parameters of pub-sub systems and they use P2P-based techniques as a means towards this end. In contrast, our goal is to enhance the P2P data sharing systems, and continuous queries (that bear similarity to pub-sub model) is a means towards that end. Second, the above pub-sub systems cannot be implemented on top of generic P2P networks; they need specialized overlays (specific topologies and/or indexing mechanisms). Contrastingly, our CoQUOS middleware does not need any distributed indexing structures, nor does it impose any topological constraints on the overlay network. Finally, the above systems are essentially pub-sub systems, and hence guaranteed notification is one of their design goals. Our system provides best-effort notification, which is in tune with design principles of unstructured P2P networks. P2P-DIET [11] supports both ad-hoc and continuous queries, however, it assumes a super peer-based overlay. In contrast, the CoQUOS system does not impose any topological constraints on the overlay.

Recently, researchers have studied random walk as a scalable and efficient alternative to flooding-based searching in unstructured P2P networks [4, 8, 9, 12]. Several variants of random walk have also been designed for this purpose [6, 23]. However, studies have reported that the performance of the random walk degrades on networks that exhibit considerable degree of node clustering. The cluster resilient random walk scheme that we have proposed for continuous query propagation attempts to alleviate this problem by favoring out-of-cluster nodes.

In short, the work presented in this paper has several unique aspects, and it addresses an important problem in the area of P2P data sharing systems.

6 Conclusions

The utility and effectiveness of P2P content distribution networks can be greatly enhanced by incorporating two capabilities, namely, mechanisms that enable individual peers to register queries and receive notifications when matching data items are added, and techniques that allow peers to advertise their data items to other interested peers. Although P2P-based pub-sub systems provide similar capabilities, developing full-fledged pub-sub systems on top of unstructured P2P networks requires complex mechanisms, which are not only costly, but are also difficult to develop and manage.

In this paper, we proposed an alternate paradigm called continuous query, which is similar to publish-subscribe model in its functionality, but provides best-effort notification service. We presented CoQUOS – lightweight middleware for supporting continuous queries and advertisements in unstructured overlay networks. The CoQUOS middleware relies upon a novel scheme for registering a continuous query at various regions of the P2P network. This scheme encompasses two unique techniques, namely cluster resilient walk mechanism for query propagation and dynamic probability technique for query registrations. We evaluated the system through a set of experiments showing the benefits and costs of the proposed schemes and techniques.

References

[1] Gnutella P2P Network. www.gnutella.com.

[2] S. Androutsellis-Theotokis and D. Spinellis. A Survey of Peer-to-Peer Content Distribution Technologies. *ACM Comput. Surv.*, 2004.

[3] G. Banavar, T. Chandra, B. Mukherjee, J. Nagarajarao, R. E. Strom, and D. C. Sturman. An Efficient Multicast Protocol for Content-Based Publish-Subscribe Systems. In *Proceedings of ICDCS 1999*, 1999.

[4] N. Bisnik and A. Abouzeid. Modeling and analysis of random walk search algorithms in P2P networks. In *Proceedings of HOT-P2P, 2005*.

[5] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, 2001.

[6] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making Gnutella-like P2P Systems Scalable. In *Proceedings of ACM SIGCOMM 2003*, 2003.

[7] P. Chirita, S. Idreos, M. Koubarakis, and W. Nejdl. Publish/Subscribe for RDF-based P2P Networks. In *Proceedings of the 1st European Semantic Web Symposium*, May 2004.

[8] C. Gkantsidis, M. Mihail, and A. Saberi. Random Walks in Peer-to-Peer Networks. In *Proceedings of the IEEE INFOCOM 2004*, 2004.

[9] C. Gkantsidis, M. Mihail, and A. Saberi. Hybrid search schemes for unstructured peer-to-peer networks. In *Proceedings of INFOCOM*, 2005.

[10] A. Gupta, O. D. Sahin, D. Agrawal, and A. E. Abbadi. Meghdoot: content-based publish/subscribe over P2P networks. In *Proceedings of Middleware 2004*, 2004.

[11] S. Idreos, M. Koubarakis, and C. Tryfonopoulos. P2P-DIET: One-Time and Continuous Queries in Super-Peer Networks. In *Proceedings of EDBT*, 2004.

[12] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *Supercomputing*, 2002.

[13] L. Ramaswamy, J. Chen, P. Parate, and A. Meka. Lightweight Support for Continuous Queries in Unstructured Overlays. Technical report, The University of Georgia, 2006.

[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A Scalable Content-Addressable Network. In *Proceedings of ACM SIGCOMM 2001*, Aug 2001.

[15] P. Reynolds and A.Vahdat. Efficient peer-to-peer keyword searching. In *Proceedings of Middleware 2003*.

[16] M. Ripeanu, A. Iamnitchi, and I. Foster. Mapping the gnutella network. *IEEE Internet Computing*, 2002.

[17] K. Sripanidkulchai. The popularity of gnutella queries and its implications on scalability, Feb 2001. Featured on O'Reilly's www.openp2p.com website.

[18] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM 2001*, Aug 2001.

[19] P. Triantafillou and I. Aekaterinidis. Content-Based Publish-Subscribe Over Structured P2P Networks. In *Proceedings of the International Workshop on Distributed Event-Based Systems (DEBS)*, 2004.

[20] S. Voulgaris, E. Riviere, A.-M. Kermarrec, and M. van Steen. Sub-2-Sub: Self-Organizing Content-Based Publish Subscribe for Dynamic Large Scale Collaborative Networks. In *Proceedings of the 5th international workshop on peer-to-peer systems*, Feb 2006.

[21] B. Yang and H. Garcia-Molina. Improving search in peer-to-peer systems. In *Proceedings of ICDCS 2002*.

[22] C. Zhang, A. Krishnamurthy, and R. Wang. Combining flexibility and scalability in a peer-to-peer publish/subscribe system. In *Middleware 2005*.

[23] M. Zhong and K. Shen. Popularity-Biased Random Walks for Peer-to-Peer Search under the Square-Root Principle. In *Proceedings of IPTPS*, 2006.