Scalable Semantic Overlay Generation for P2P-Based Digital Libraries

Christos Doulkeridis[1], Kjetil Nørvåg[2], and Michalis Vazirgiannis[1]

[1]Dept. of Informatics, AUEB, Athens, Greece
{cdoulk,mvazirg}@aueb.gr
[2]Dept. of Computer Science, NTNU, Trondheim, Norway
Kjetil.Norvag@idi.ntnu.no

Abstract. The advent of digital libraries along with the tremendous growth of digital content call for distributed and scalable approaches for managing vast data collections. Peer-to-peer (P2P) networks emerge as a promising solution to delve with these challenges. However, the lack of global content/topology knowledge in an unstructured P2P system demands unsupervised methods for content organization and necessitates efficient and high quality search mechanisms. Towards this end, Semantic Overlay Networks (SONs) have been proposed in the literature, and in this paper, an unsupervised method for decentralized and distributed generation of SONs, called DESENT, is proposed. We prove the feasibility of our approach through analytical cost models and we show through simulations that, when compared to flooding, our approach improves recall by more than 3-10 times, depending on the network topology.

1 Introduction

The advent of digital libraries along with the tremendous growth of digital content call for distributed and scalable approaches for managing vast data collections. Future digital libraries will enable citizens to access knowledge any time/where, in a friendly, multi-modal, efficient and effective way. Reaching this vision requires development of new approaches that will significantly reform the current form of digital libraries. Key issues in this process are [9]: the system architecture and the information access means. With respect to system architecture, peer-to-peer (P2P) is identified as a topic of primary interest, as P2P architectures allow for loosely-coupled integration of information services and sharing of information/knowledge [1,6,11].

In this paper, we present a scalable approach to P2P document sharing and retrieval. Because scalability and support for semantics can be difficult in structured P2P systems based on DHTs, we instead base our approach on *unstructured P2P networks*. Such systems, in their basic form, suffer very high search costs, in terms of both consumed bandwidth and latency, so in order to be useful for real applications, more sophisticated search mechanisms are required. We solve this problem by employing *semantic overlay networks* (SONs) [5], where peers containing related information are connected together in separate overlay networks. If SONs have been created, queries can be forwarded to only those peers containing documents that satisfy the constraints of the query context, for example based on topic, user profiles or features extracted from previous queries.

One of the problems of SONs is the actual construction of these overlays, because in a P2P context there is a lack of knowledge of both global content and network topology. In a P2P architecture, each peer is initially aware only of its neighbors and their content. Thus, finding other peers with similar contents, in order to form a SON, becomes a tedious problem. This contrasts to a centralized approach, where all content is accessible to a central authority, and clustering becomes a trivial problem, in the sense that only the clustering algorithm (and its input parameter values) determines the quality of the results.

The contribution of this paper is a *distributed* and *decentralized* method for hierarchical SON construction (DESENT) that provides an efficient mechanism for search in unstructured P2P networks. Our strategy for creating SONs is based on clustering peers based on their content similarity. This is achieved by a recursive process that starts on the individual peers. Through applying a clustering algorithm on the documents stored at the peer, one or more feature vectors are created for each peer, essentially one for each topic a peer covers. Then representative peers, each responsible for a number of peers in a *zone* are selected. These peers, henceforth called *initiators*, will collect the feature vectors from the members of the zone and use these as basis for the next level of clustering. This process is applied recursively, until we have a number of feature vectors covering all available documents.

The organization of the rest of this paper is as follows. In Section 2, we give an overview of related work. In Section 3, we present our method for creating SONs that can be used in the search process (Section 4). In Section 5, we use analytical cost models to study the cost and the time required for overlay creation, while, in Section 6, we present the simulation results. Finally, in Section 7, we conclude the paper.

2 Related Work

Several techniques have been proposed that can improve search in unstructured P2P systems [2,8], including techniques for improved routing that give a direction towards the requested document, like routing indices [4], and connectivity-based clustering that creates topological clusters that can be used as starting points for flooding [12]. An approach to improve some of the problems of Gnutella-like systems [2], is to use a super-peer architecture [15], which can be also used to realize a hierarchical summary index, as described in [13].

The concept of semantic overlay networks (SONs) [5] is about directing searches only to a specific subset of peers with content relevant to the query. The advantage of this approach is that it reduces the flooding cost in the case of unstructured systems. Crespo and Garcia-Molina [5] essentially base their approach on partly pre-classified documents that only consist of information about the song contained in a particular file. Also they do not provide any other algorithm for searching, other than flooding. In order to be useful in a large system, unsupervised and decentralized creation of SONs is necessary, as well as efficient routing to the appropriate SON(s). The DESENT approach described in our paper solves these issues.

Although several papers describe how to use SON-like structures for P2P content search [3,10], little work exists on the issue of how to actually create SONs in an unsupervised, decentralized and distributed way in unstructured networks. Distributed clustering in itself is considered a challenge demanding for efficient and effective solutions. In [14], a P2P architecture where nodes are logically organized into a fixed number of clusters is presented. The main focus of the paper is fairness with respect to the load of individual nodes. In contrast to our approach, the allocation of documents to clusters is done by classification, it is not unsupervised, and clusters are not hierarchical. We believe that current research in P2P digital libraries [1,6,11] can benefit from the merits of our approach.

3 Overlay Network Creation

In this section, we describe SON generation, assuming peers storing digital content and being connected in an unstructured P2P network. Each peer represents a digital library node and in this paper we focus on peers that store documents, though other data representations can also be supported. The approach is based on creating local zones of peers, forming semantic clusters based on data stored on these peers, and then merging zones and clusters recursively until global zones and clusters are obtained.

3.1 Decentralized and Distributed Cluster Creation

The peer clustering process is divided into 5 phases: 1) local clustering, 2) zone initiator selection, 3) zone creation, 4) intra-zone clustering, and 5) inter-zone clustering.

Phase 1: Local Clustering. In the process of determining sites that contain related documents, *feature vectors* are used instead of the actual documents because of the large amounts of data involved. A feature vector F_i is a vector of tuples, each tuple containing a feature (word) f_i and a weight w_i. The feature vectors are created using a feature extraction process (more on the feature extraction process in section 6). By performing clustering of the document collection at each site, a set of document clusters is created, each cluster represented by a feature vector.

Phase 2: Initiator Selection. In order to be able to create zones, a subset of the peers have to be designated the role of *zone initiators* that can perform the zone creation process and subsequently initiate and control the clustering process within the zone.

The process of choosing initiators is completely distributed and ideally would be performed at all peers concurrently, in order to have approximately S_Z peers in each zone[1]. However, this concurrency is not necessary, since the use of zone partitioning at the next phase eliminates the danger of excessive zone sizes.

Assuming the IP of a peer P_i is IP_{P_i} and the time is T (rounded to nearest t_a[2]), a peer will discover that it is an initiator if $(IP_{P_i} + T)\ MOD\ S_Z = 0$. The aim of the function is to select initiators that are uniformly spread out in the network and an appropriate number of initiators relative to the total number of peers in the network.

[1] In order to avoid some initiators being overloaded, the aim is to have as uniform zone sizes as possible. Note that although uniform zone size and having initiator in the center of the zone are desired for load-balancing reasons, this is not crucial for the correctness or quality of the overlay construction.

[2] Assuming that each peer has a clock that is accurate within a certain amount of time t_a, note that DESENT itself can be used to improve the accuracy.

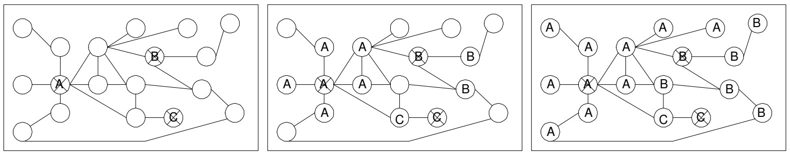

Fig. 1. Step-wise zone creation given the three initiators A, B, and C.

By including time in the function we ensure that we obtain different initiators each time the clustering algorithm is run. This tackles the problem of being stuck with faulty initiators, as well as reduces the problem of permanent cheaters.

If no initiator is selected by the above strategy, this will be discovered from the fact that the subsequent zone creation phase is not started within a given time (i.e., no message received from an initiator). In this case, a universal decrease of the modulo-parameter is performed, by dividing by an appropriate prime number, as many times as necessary, in order to increase the chance of selecting (at least) one peer at the next iteration.

Phase 3: Zone Creation. After a peer P_i has discovered that it is an initiator, it uses a probe-based technique to create its zone. An example of zone creation is illustrated in Fig. 1. This zone creation algorithm has a low cost wrt. to number of messages (see Section 5), and in the case of excessive zone sizes, the initiator can decide to partition its zone, thus sharing its load with other peers. When this algorithm terminates, 1) each initiator has assembled a set of peers Z_i and their capabilities, in terms of resources they possess, 2) each peer knows the initiator responsible for its zone and 3) each initiator knows the identities of its neighboring initiators. An interesting characteristic of this algorithm is that it ensures that all peers in the network will be contacted, as long as they are connected to the network. This is essential, otherwise there may exist peers whose content will never be retrieved. We refer to the extended version of this paper for more details on initiator selection and zone creation [7].

Phase 4: Intra-zone Clustering. After the zones and their initiators have been determined, global clustering starts by collecting feature vectors from the peers (one feature vector for each cluster on a peer) and creating clusters based on these feature vectors:

1. The initiator of each zone i sends probe messages *FVecProbe* to all peers in Z_i.
2. When a peer P_i receives a *FVecProbe* it sends its set of feature vectors $\{F\}$ to the initiator of the zone.
3. The initiator performs clustering on the received feature vectors. The result is a set of clusters represented by a new set of feature vectors $\{F_i\}$, where an F_i consists of the top-k features of cluster C_i. Note that a peer can belong to more than one cluster. In order to limit the computations that have to be performed in later stages at other peers, when clusters from more than one peer have to be considered, the clustering should result in at most N_C^0 such basic clusters (N_C^0 is controlled by the clustering algorithm). The result of this process is illustrated in the left part of Fig. 2.
4. The initiator selects a representative peer R_i for each cluster, based on resource information that is provided during Phase 3, like peer bandwidth, connectivity, etc. One of the purposes of a representative peer is to represent a cluster at search time.

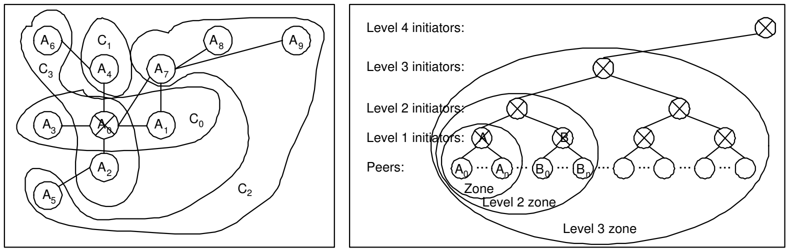

Fig. 2. Left: Possible result of intra-zone clustering of zone A, resulting in the four clusters C_0, C_1, C_2, and C_3. Right: Hierarchy of zones and initiators.

5. The result kept at the initiator is a set of cluster descriptions (CDs), one for each cluster C_i. A CD consists of the cluster identifier C_i, a feature vector F_i, the set of peers $\{P\}$ belonging to the cluster, and the representative R of the cluster, i.e., CD$_i$ $= (C_i, F_i, \{P\}, R)$. For example, the CD of cluster C_2 in Fig. 2 (assuming A_7 is the cluster representative) would be CD$_2$ $= (C_2, F_2, \{A_5, A_7, A_8, A_9\}, A_7)$.
6. Each of the representative peers are informed by the initiator about the assignment and receive a copy of the CDs (of *all* clusters in the zone). The representatives then inform peers on their cluster membership by sending them messages of the type (C_i, F_i, R).

Phase 5: Inter-zone Clustering. At this point, each initiator has identified the clusters in its zone. These clusters can be employed to reduce the cost and increase the quality of answers to queries involving the peers in one zone. However, in many cases peers in other zones will be able to provide more relevant responses to queries. Thus, we need to create an overlay that can help in routing queries to clusters in remote zones. In order to achieve this, we recursively apply merging of zones to larger and larger super-zones, and at the same time merge clusters that are sufficiently similar into super-clusters: first a set of neighboring zones are combined to a super-zone, then neighboring super-zones are combined to a larger super-zone, etc. The result is illustrated in the right part of Fig. 2 as a hierarchy of zones and initiators. Note that level-i initiators are a subset of the level-$(i-1)$ initiators.

This creation of the inter-zone cluster overlay is performed as follows:

1. From the previous level of zone creation, each initiator maintains knowledge about its neighboring zones (and their initiators). Thus, the zones essentially form a zone-to-zone network resembling the P2P network that was the starting point.
2. A level-i zone should consist of a number of neighboring level-$(i-1)$ zones, on average $|SZ|$ in each (where SZ denotes a set of zones, and $|SZ|$ the number of zones in the set). This implies that $\frac{1}{|SZ|}$ of the level-$(i-1)$ initiators should be level-i initiators. This is achieved by using the same technique for initiator selection as described in Phase 2, except that in this case only peers already chosen to be initiators at level-$(i-1)$ in the previous phase are eligible for this role.
3. The level-i initiators create super-zones using the algorithm of Phase 3. In the same way, these level-i initiators will become aware of their neighboring super-zones.

4. In a similar way to how feature vectors were collected during the basic clustering, the approximately $N_C|SZ|$ CDs created at the previous level are collected by the level-i initiator (where N_C denotes the number of clusters per initiator at the previous level). Clustering is performed again and a set of super-clusters is generated. Each of the newly formed super-clusters is represented by top-k features produced by merging the top-k feature vectors of the individual clusters. The result of cluster merging is a set of super-clusters. A peer inside the super-cluster (not necessarily one of the representatives of the cluster) is chosen as representative for the super-cluster. The result is a new set of CDs, CD$_i = (C_i, F_i, \{P\}, R)$, where the set of peers {P} contains the representatives of the clusters forming the base of the new super-cluster.
5. The CDs are communicated to the appropriate representatives. The representatives of the merged clusters (the peers in $\{P\}$ in the new CDs) are informed about the merging by the super-cluster representative, so that all cluster representatives know about both their representatives *below* as well as the representative *above* in the hierarchy. Note that although the same information could be obtained by traversing the initiator/super-initiator hierarchy, the use of cluster representatives distributes the load more evenly and facilitates efficient searching.

This algorithm terminates when only one initiator is left, i.e., when an initiator has no neighbors. Unlike the initiators at the previous levels that performed clustering operations, the only purpose of the final initiator is to decide the level of the final hierarchy. The aim is to have at the top level a number of initiators that is large enough to provide load-balancing and resilience to failures, but at the same time low enough to keep the cost of exchanging clustering information between them during the overlay creation to a manageable level. Note that there can be one or more levels below the top-level initiator that have too few peers. The top-level peer probes level-wise down the tree in order to find the number of peers at each level until it reaches level j with appropriate number *min*$_F$ of peers. The level-j initiators are then informed about the decision and they are given the identifiers of the other initiators at that level, in order to send their CDs to them. Finally, all level-j initiators have knowledge about the clusters in zones covered by the other level-j initiators.

3.2 Final Organization

To summarize, the result of the zone- and cluster-creation process are two hierarchies:

Hierarchy of peers: Starting with individual peers at the bottom level, forming zones around the initiating peer which acts as a zone controller. Neighboring zones recursively form super-zones (see right part of Fig. 2), finally ending up in a level where the top of the hierarchies have replicated the cluster information of the other initiators at that level. This is a forest of trees. The peers maintain the following information about the rest of the overlay network: 1) Each peer knows its initiator. 2) A level-1 initiator knows the peers in its zone as well as the level-2 initiator of the super-zone it is covered by. 3) A level-i initiator (for $i > 1$) knows the identifiers of the level-$(i-1)$ initiators of the zones that constitute the super-zone as well as the level-$(i+1)$ initiator of the super-zone it is covered by. 4) Each initiator knows all cluster representatives in its zone.

Hierarchy of clusters: Each peer is member of one or more clusters at the bottom level. Each cluster has one of its peers as representative. One or more clusters constitute a super-cluster, which again recursively form new super-clusters. At the top level a number of global clusters exist. The peers store the following information about the cluster hierarchy: 1) Each peer knows the cluster(s) it is part of, and the representative peers of these clusters. 2) A representative also knows the identifiers of the peers in its cluster, as well as the identifier of the representative of the super cluster it belongs to. 3) A representative for a super-cluster knows the identifier of the representative at the level above as well as the representatives of the level below.

3.3 Peer Join

A peer P_J that joins the network first establishes connection to one or more peers as part of the basic P2P bootstrapping protocol. These neighbors provide P_J with their zone initiators. Through one of these zone initiators, P_J is able to reach one of the top-level nodes in the zone hierarchy and through a search downwards find the most appropriate lowest-level cluster, which P_J will then subsequently join. Note that no reclustering will be performed, so after a while a cluster description might not be accurate, but that cannot be enforced in any way in a large-scale, dynamic peer-to-peer system, given the lack of total knowledge. However, the global clustering process is performed at regular intervals and will then create a new clustering that reflects also the contents of new nodes (as well as new documents that have changed the individual peer's feature vectors). This strategy considerably reduces the maintenance cost, in terms of communication bandwidth compared with incremental reclustering, and also avoids the significant cost of continuous reclustering.

4 Searching

In this section we provide an overview of query processing in DESENT. A query Q in the network originates from one of the peers P, and it is continually expanded until satisfactory results, in terms of number and quality, have been generated. All results that are found as the query is forwarded are returned to P. Query processing can terminate at any of the steps below if the result is satisfactory:

1. The query is evaluated locally on the originating peer P.
2. A peer is a member of one or more clusters C_i. The C_i which has the highest similarity $sim(Q, C_i)$ with the query is chosen, and the query is sent to and evaluated by the other peers in this cluster.
3. Q is sent to one of the top-level initiators (remember that each of the top-level initiators knows about all the top-level clusters). At this point we employ two alternatives for searching:
   (a) The most appropriate top-level cluster is determined based on a similarity measure, and Q is forwarded to the representative of that cluster. Next, Q is routed down the cluster hierarchy until the query is actually executed at the peers in a lowest-level cluster. The path is chosen based on highest $sim(Q, C_i)$ of the actual sub-clusters of a level-i cluster. If the number of results is insufficient, then backtracking is performed in order to extend the query to more clusters.

	Parameter	Default Value		Parameter	Default Value
B	Minimum bandwidth available	1 KB/s	N_i	# of peers/zones at level i	$\frac{N_P}{(S_Z)^i}$
D_0	Avg. # of neighbors at level 0	4	N_P	Total # of peers in the network	1000000
D_i	Avg. # of neighbors at level i	S_Z	r	Max zone radius	20
L	# of initiator levels	$\lfloor \log_{S_Z} N_P \rfloor$	S_{CD}	Size of a CD	$\approx 1.5 S_F$
min_F	Min. # of trees in top-level forest	$S_Z/4$	S_F	Size of feature vector	200 bytes
N_C^0	# of clusters per peer	10	S_M	Size of packet overhead	60 bytes
N_C^i	# of clusters per level-i initiator	100	S_Z	Avg. zone size	100
N_F	# of trees in top-level forest	$> S_Z/4$	t_a	Time between synch. points	60 seconds

Table 1. Parameters and default values used in the cost models.

(b) All top-level clusters that have some similarity $sim(Q, C_i) > 0$ to the query Q are found and the query is forwarded to *all* cluster representatives. The query is routed down at *all* paths of the cluster hierarchy until level-0. Practically, all subtrees that belong to a matching top-level cluster are searched extensively.

The first approach reduces query latency, since the most relevant subset of peers will be identified with a small cost of messages. However, the number of returned documents will probably be restricted, since the search will focus on a local area only. This approach is more suitable for top-k queries. The second approach can access peers residing at remote areas (i.e. remote zones), with acceptable recall, however this results in a larger number messages. It is more suitable for cases when we are interested in the completeness of the search (retrieval of as many relevant documents as possible). In the following, we provide simulation results only for the second scenario, since we are mainly interested in testing the recall of our approach.

5 Feasibility Analysis

We have studied the feasibility of applying DESENT in a real-world P2P system through analytical cost models. Due to lack of space, we present here only the main results of the analytical study, whereas the actual cost models are described in detail in the extended version of this paper [7]. The parameters and default values used in the cost models are summarized in Table 1. These are typical values (practically size and performance) or values based on observations and conclusions from simulations.

A very important concern is the burden the DESENT creation imposes on participating nodes. We assume that the communication cost is the potential bottleneck and hence the most relevant metric, and we consider the cost of creating DESENT acceptable if the cost it imposes is relatively small compared to the ordinary document-delivery load on a web server.

In studying the feasibility of DESENT, it is important that the *average* communication cost for each peer is acceptable, but most important is the *maximum* cost that can be incurred for a peer, i.e., the cost for the initiators on the top level of the hierarchy. In order to study the maximum cost C_M for a particular peer to participate in the creation of the overlay network, both received and sent data should be counted because both pose a burden on the peer. Fig. 3 (left) illustrates C_M for different values of N_P

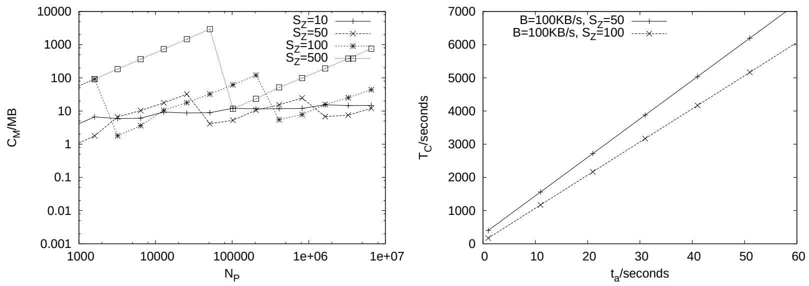

Fig. 3. Left: maximum cost of participation in overlay network creation for different values of network size N_P and zone size S_Z. Right: Time T_C to create DESENT as a function of t_a for different zone sizes and bandwidths.

and zone size S_Z. We see that a large zone size results in higher cost, but with very high variance. The situations in which this happens, is when the number of top-level peers is just below the *min*$_F$ threshold so that the level below will be used as top level instead. With a large zone size this level will contain a large number of peers, and the final exchange of clusters information between the roots of this forest will be expensive. However, in practice this could be solved by merging of zones at this level. Regarding the maximum cost, if we consider a zone size of $S_Z = 100$, the maximum cost is just above 100 MB. Compared with the load of a typical web server, which is some GB of delivered documents per day, [3] this is acceptable even in the case of daily reclustering. However, considering the fact that the role of the upper-level initiators changes every time the overlay network is created, it could even be feasible to perform this clustering more often. In addition to the cost described above, there will also be a certain cost for maintaining replicas and peer dynamics in the network. However, this cost will be relatively small compared to the upper-level exchange of CDs.

In order to ensure freshness of the search results, it is important that the duration of the DESENT creation itself is not too long. The results, illustrated in Fig. 3 (right), show the time needed to create DESENT for different values of maximum assumed clock deviation, zone size S_Z, and minimum available bandwidth for DESENT participation B. For typical parameter values and $t_a = 30s$, the time needed to construct the DESENT overlay network is between 3000 and 4000 seconds, i.e., approximately one hour. This means that the DESENT creation could run several times a day, if desired. An important point is that even if the construction takes a certain time, the average load the construction imposes on peers will be relatively low. Most of the time is used to ensure that events are synchronized, without having to use communication for this purpose. Regarding values of parameters, it should be stressed that the actual *number of peers* has only minimal impact on the construction time, because the height of the tree is the important factor, and this increases only logarithmically with the number of peers.

[3] Using a web server in our department as example, it delivers in the order of 4 GB per day, and a large fraction of this data is requested by search engines crawling the web.

6 DESENT Simulation Results

We have developed a simulation environment in Java, which covers all intermediate phases of the overlay network generation as well as the searching part. We ran all our experiments on Pentium IV computers with 3GHz processors and 1-2GB of RAM.

At initialization of the P2P network, a topology of N_P interconnected peers is created. We used the GT-ITM topology generator[4] to create random graphs of peers (we also used power-law topologies with the same results, due to the fact that the underlying topology only affects the zone creation phase), and our own SQUARE topology, which is similar to GT-ITM, only the connectivity degree is constant and neighboring peers share 3-5 common neighbors, i.e., the network is more dense than GT-ITM. A collection of N_D documents is distributed to peers, so that each peer retains N_D/N_P distinct documents. Every peer runs a clustering algorithm on its local documents resulting in a set of initial clusters. In our experiments we chose the Reuters-21578 text categorization test collection,[5] and we used 8000 pre-classified documents that belong to 60 distinct categories, as well as a different setup of 20000 documents. We tried different experimental setups with 2000, 8000 and 20000 peers. We then performed feature extraction (tokenization, stemming, stop-word removal and finally keeping the top-k features based on their TF/IDF[6] value and kept a feature vector of top-k features for each document as a compact document description). Thus, each document is represented by a top-k feature vector. Initiators retrieve the feature vectors of all peers within their zone, in order to execute intra-zone clustering. We used hierarchical agglomerative clustering (HAC) to create clusters of documents. Clustering is based on computing document similarities and merging feature vectors, by taking the union of the clusters' features and keeping the top-k features with higher TF/IDF values. We used the cosine similarity with parameter the similarity threshold T_s for merging. Clusters are created by grouping together sufficiently similar documents and each cluster is also represented by a top-k feature vector. Obviously, other clustering algorithms, as well as other similarity measures can be used.

6.1 Zone Creation

We studied the average zone size after the zone creation phase at level 1. The network topology consists of $N_P = 20000$ peers, each having 10 neighbors on average and $S_Z = 100$. We run the experiment with and without the zone partitioning mechanism. The simulations brought out the value of zone partitioning, since this mechanism keeps all zones smaller than S_Z, while most are of sizes $50 - 100$. However, when there is no zone partitioning, about 30% of the total zones have sizes greater than S_Z, and some are twice larger than S_Z, thus imposing a cumbersome load on several initiators.

[4] `http://www.cc.gatech.edu/projects/gtitm/`

[5] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

[6] Notice that the inverse document frequency (IDF) is not available, since no peer has global knowledge of the document corpus, so we use the TF/IDF values produced on each peer locally, taking only the local documents into account.

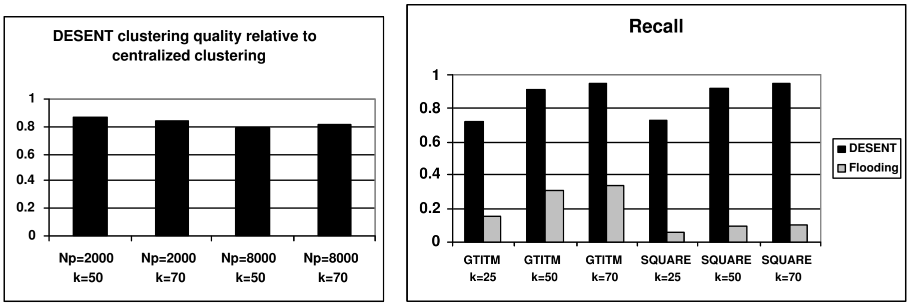

Fig. 4. Simulation results: Cluster quality, compared to centralized clustering, for different network sizes and values of k (left), and average recall compared to normalized flooding using the same number of messages (right).

6.2 Clustering Results Quality

Measuring the quality of the DESENT clustering results is essential for the value of the approach. As clustering quality in our context, we define the similarity of the results of our clustering algorithm (C_i), with respect to an optimal clustering (K_j). We used in our experiments the F-measure as a cluster quality measure. F-measure ranges between 0 and 1, with higher values corresponding to better clustering.

We compare the clustering quality of our approach to the centralized clustering results. The average values of DESENT F-measure relative to centralized clustering are illustrated in the left part of Fig 4, and show that DESENT achieves high clustering quality. Also note that the results exhibit a relatively stable behavior as the network size increases. This indicates that DESENT scales well with the number of participating peers. This conveys that the proposed system achieves high quality in forming SONs despite of the lack of global knowledge and the high distribution of the content.

6.3 Quality and Cost of Searching

In order to study the quality of searching in DESENT, we consider as baseline the search that retrieves all documents that contain all keywords in a query. We measure the searching quality using recall, representing the percentage of the relevant documents found. Note that, for the assumed baseline, precision will always be 100% in our approach, since the returned documents will always be relevant, due to the exact matching of all keywords. We generated a synthetic query workload consisting of queries with term count average 2.0 and standard deviation 1.0. We selected query terms from the documents randomly (ignoring terms with frequency less than 1%). The querying peer was selected randomly.

In the right part of Fig. 4, we show the average recall of our approach compared to normalized flooding using the same number of messages for different values of k, for the GT-ITM topology and the SQUARE topology for 8000 peers. Normalized flooding [8] is a variation of naive flooding that is widely used in practice, in which each peer forwards a query to d neighbors, instead of all neighbors, where d is usually the minimum connectivity degree of any peer in the network. The chart shows that with the same number of messages, our approach improves recall by more than 3-5 times for GT-ITM, and more than 10 for SQUARE, compared to normalized flooding. Furthermore, the absolute recall values increase with k, since more queries can match the enriched (with more features) cluster descriptions. Also notice that our approach presents the same recall independent of the underlying network topology.

7 Conclusions and Further Work

In this paper, we have presented algorithms for distributed and decentralized construction of hierarchical SONs, for supporting searches in a P2P-based digital library context. Future work includes performance and quality measurement of the search algorithm using large document collections, studying the use of other clustering algorithms as well as the use of caching techniques and ranking to increase efficiency.

Acknowledgments. The authors would like to thank George Tsatsaronis and Semi Koen for their help in preparing the feature extraction and clustering modules.

References

1. W.-T. Balke, W. Nejdl, W. Siberski, and U. Thaden. DL meets P2P - Distributed Document Retrieval based on Classification and Content. In *Proceedings of ECDL'2005*, 2005.
2. Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making Gnutella-like P2P Systems Scalable. In *Proceedings of SIGCOMM'03*, 2003.
3. E. Cohen, H. Kaplan, and A. Fiat. Associative Search in Peer-to-Peer Networks: Harnessing Latent Semantics. In *Proceedings of INFOCOM'03*, 2003.
4. A. Crespo and H. Garcia-Molina. Routing Indices for Peer-to-Peer Systems. In *Proceedings of ICDCS'2002*, 2002.
5. A. Crespo and H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. Technical report, Stanford University, 2002.
6. H. Ding and I. Sølvberg. Choosing Appropriate Peer-to-Peer Infrastructure for your Digital Libraries. In *Proceedings of ICADL'2005*, 2005.
7. C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. DESENT: Decentralized and Distributed Semantic Overlay Generation in P2P Networks. Technical report, AUEB, 2005 `http://www.db-net.aueb.gr/index.php/publications/technical_reports/`.
8. C. Gkantsidis, M. Mihail, and A. Saberi. Hybrid Search Schemes for Unstructured Peer-to-Peer Networks. In *Proceedings of INFOCOM'05*, 2005.
9. Y. Ioannidis, H.-J. Schek, and G. Weikum, editors. *Proceedings of the 8th International Workshop of the DELOS Network of Excellence on Digital Libraries on Future Digital Library Management Systems (System Architecture & Information Access)*, 2005.
10. X. Liu, J. Wang, and S. T. Vuong. A Category Overlay Infrastructure for Peer-to-Peer Content Search. In *Proceedings of IPDPS'05*, 2005.
11. H. Nottelmann and N. Fuhr. Comparing Different Architectures for Query Routing in Peer-to-Peer Networks. In *Proceedings of ECIR'2006*, 2006.
12. L. Ramaswamy, B. Gedik, and L. Liu. Connectivity based Node Clustering in Decentralized Peer-to-Peer Networks. In *Proceedings of P2P'03*, 2003.
13. H. T. Shen, Y. Shu, and B. Yu. Efficient Semantic-based Content Search in P2P Network. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):813–826, 2004.
14. P. Triantafillou, C. Xiruhaki, M. Koubarakis, and N. Ntarmos. Towards High Performance Peer-to-Peer Content and Resource Sharing Systems. In *Proceedings of CIDR'03*, 2003.
15. B. Yang and H. Garcia-Molina. Designing a Super-Peer Network. In *Proceedings of ICDE'03*, 2003.