A Hybrid Approach for Estimating Document Frequencies in Unstructured P2P Networks

Robert Neumayer, Christos Doulkeridis, and Kjetil Nørvåg

Norwegian University of Science and Technology
Sem Sælands vei 7-9, 7491, Trondheim, Norway
{*neumayer,cdoulk,noervaag*}*@idi.ntnu.no*

Abstract

Scalable search and retrieval over numerous web document collections distributed across different sites can be achieved by adopting a peer-to-peer (P2P) communication model. Terms and their document frequencies are the main components of text information retrieval and as such need to be computed, aggregated, and distributed throughout the system. This is a challenging problem in the context of unstructured P2P networks, since the local document collections may not reflect the global collection in an accurate way. This might happen due to skews in the distribution of documents to peers. Moreover, central assembly of the total information is not a scalable solution due to the excessive cost of storage and maintenance, and because of issues related to digital rights management. In this paper, we present an efficient hybrid approach for aggregation of document frequencies using a hierarchical overlay network for a carefully selected set of the most important terms, together with gossip-based aggregation for the remaining terms in the collections. Furthermore, we present a cost analysis to compute the communication cost of hybrid aggregation. We conduct experiments on three document collections, in order to evaluate the quality of the proposed hybrid aggregation.

Keywords: Peer-to-peer, distributed information retrieval, distributed aggregation

1. Introduction

Text search in distributed document collections is an important challenge in connection with today's usage of the Internet, or more generally, networked services. Both the number of users and the number of available documents – mutually influencing each other, e.g., when user-generated content is considered – have been growing at impressive rates for years. However, private use is not the only area with a strong need for distributed text search and mining. Professional institutions such as libraries, museums, or companies increasingly use their documents in digital form.

Modern applications are increasingly deployed over widely distributed data sources and each of them stores vast amounts of data, a development partly driven by the growth of the web itself. Web information retrieval settings are a good example for such architectures, as they contain large document collections stored at disparate locations. Central assembly of the total information is neither feasible, as digital rights do not allow replication of documents, nor effective, since the cost of storing and maintaining this information is excessive. In order to achieve interoperability and intercommunication, there exists a need for loosely-coupled architectures that facilitate searching over the complete information available. Peer-to-peer (P2P) networks constitute a scalable solution for managing highly distributed document collections. As a consequence, such systems have often been used in web information retrieval and web search settings [4, 7, 8, 23, 27].

One of the main problems in distributed retrieval lies in the difficulty of providing a qualitative ranking of documents with respect to user queries. In this context, the baseline or reference is the centralized case. At the same time, performance and scalability considerations play a vital role in the development and applicability of such a widely distributed system. Thus, the important problem in the context of unstructured P2P networks is to provide a comprehensive ranking of terms (and documents). Information about how many documents a term appears in (the so called document frequency of a given term) is vital for the task of searching documents and ranking the results of these searches according to popular information retrieval ranking models. As such, document frequency estimation is one of the main components of a search system and it is particularly difficult in the distributed context. Clearly, exchanging all terms and their respective document frequencies would be a solution, however the cost is prohibitive, even for modest network sizes and medium-sized document collections, and even more so for dynamically evolving collections. Therefore, we need a pre-selection of terms at the peer level to evaluate the usefulness of terms locally. The more flexible an approach is in handling held back terms, the more stable it is with respect to cheating or withholding of information by single peers. This aggregation process must work well without consuming excessive bandwidth, regardless of the size of the network topology.

In general, two alternatives exist for performing aggregation in unstructured P2P networks: 1) building a hierarchical overlay network that enables *hierarchical aggregation*, and 2) adopting a *gossip-based aggregation* protocol. Each approach has its own merits and shortcomings. Hierarchical aggregation is efficient, fast and results in accurate values. Gossip-based aggregation is simple, scalable and robust to peer failures. However, it only provides probabilistic guarantees for the accuracy of aggregation and induces higher communication cost. Motivated by this discussion, we propose a hybrid approach for aggregation of term frequencies that combines hierarchical and gossip-based aggregation, thus sharing their advantages.

The hierarchical overlay network is formed in a self-organizing manner, which enables efficient aggregation of information. Then, carefully selected terms and their corresponding frequencies from each peer are pushed upwards in the hierar-

chy. A gossip-based aggregation protocol is employed to estimate the document frequency of less frequent terms by local periodic communication. In order to obtain the final results, the estimates resulting from the two approaches are merged and can be used for ranking documents or other information retrieval tasks.

For the hierarchical aggregation only a relatively small number of terms from a prohibitively large overall set of terms are selected. The remaining low-frequency terms are aggregated using gossiping. An interesting related issue is how to perform this term selection at individual peer level independently of other peers' contents, and we investigate the impact of term selection techniques in this context. Hence, the main contributions of this work are:

1. We present an approach for hierarchical aggregation that can be used to estimate with high accuracy the document frequencies of carefully selected terms, without assembling all information at a central location.
2. We complement the hierarchical aggregation approach with gossip-based aggregation, in order to estimate the document frequency of the remaining terms.
3. We develop a cost model that we use to show that our approach is scalable with respect to communication costs.
4. We conduct an experimental evaluation on three document collections demonstrating the applicability and scalability of the approach, and we investigate the accuracy of the aggregated information.

The work presented in this paper is based on techniques and experiments introduced in [19]. We now present additional techniques to handle low-frequency terms as well as the hybrid method for frequency estimation. In addition, we extend our cost model to support the novel hybrid aggregation method. We further perform new experiments including a new quality measure and retrieval experiments with an additional large-scale collection.

The remainder of this paper is structured as follows: in Section 2, we provide an overview of relevant work done in related areas. We then introduce preliminaries such as basic term selection approaches and aggregation in P2P networks in Section 3. The details of the hybrid estimation approach and the underlying hierarchical aggregation together with gossip-based elements are described in Section 4. We present a cost model for assessing the communication cost in Section 5. The experimental setup as well as evaluation in terms of document retrieval and ranking are presented in Section 6. Finally, in Section 7, we draw conclusions and give an outlook on future work.

2. Related Work

In this section, we provide an overview of research efforts that are deemed relevant to this paper. First, we provide a brief overview of related work in distributed information retrieval (DIR). We then present the most prominent approaches on aggregation in large-scale distributed systems. Furthermore, we

describe existing work in the areas of P2P document retrieval and P2P-based digital libraries (DL).

2.1. Distributed Information Retrieval (DIR)

DIR has advanced to a mature research area dealing with querying multiple, geographically distributed information repositories. Both term weighting and normalization are identified as major problems in dynamic scenarios [32], for both require global document frequency information. Viles and French study the impact of document allocation and collection-wide information in distributed archives [31]. They observe that even for a modest number of sites, dissemination of collection-wide information is necessary to maintain retrieval effectiveness, but that the amount of disseminated information can be relatively low. In a smaller scale distributed system, it is possible to use a dedicated server for collecting accurate term-level global statistics [16]. However, this approach is clearly not appropriate for large-scale systems.

In [33], the authors examine the estimation of global term weights (such as the document frequency of a term, i.e. the number of documents a term occurs in for a given collection) in information retrieval scenarios where a global view of the collection is not available. Two alternatives are studied: either sampling documents or using a reference corpus independent of the target retrieval collection. In addition, the possibility of pruning term lists based on frequency is evaluated. The results show that very good retrieval performance can be reached when just the most frequent terms of a collection (an extended stop word list) are known, and all terms which are not in that list are treated equally. The paper does not consider how to actually determine (collect) and distribute this information.

Query-based sampling is applied to the resource selection problem in DIR in [1, 2]. Predictive Likelihood is used to assess the adequacy of an acquired resource description, as opposed to other approaches, where a fixed number of samples is drawn for each collection. The authors show that their technique minimized overheads while maintaining selection performance.

In this paper, we implicitly study the effects of different term pre-selection methods on distributed document collections over an unstructured P2P network, even though its dynamic aspects are not the main concern in DIR research. Also, our experiments are specifically designed to show the effects of unequally distributed collections, which is a common case in DIR settings.

2.2. Aggregation in Large-Scale Distributed Systems

Aggregation of information in distributed systems is a challenging issue, thus it has attracted the attention of several research initiatives. One obvious method is to employ a hierarchy of nodes to perform hierarchical aggregation at intermediate levels. SDIMS [35] uses tree-based aggregation over a Distributed Hash Table (DHT) infrastructure, in order to provide a generic aggregation mechanism for large-scale systems. For other tree-based aggregation efforts we refer to Willow [30] and SOMO [37].

Gossip-based aggregation [11, 12] aims to provide a protocol for network-based aggregation in a completely decentralized manner. The actual process of aggregation is achieved through periodic interactions (also known as cycles) among nodes, which exchange aggregated values. Gossiping protocols for information aggregation also provide theoretical properties for convergence within a logarithmic number of steps with respect to the network size.

There also exist other systems that combine gossip-based aggregation with hierarchical aggregation. Probably the most well-known framework in this category is Astrolabe [29], which provides a continuously running aggregation mechanism. The purpose of aggregation is to compute aggregate values of a particular resource of interest (such as number of copies of a specific file) in a network-wide context. While Astrolabe focuses on maintaining aggregate values at any time, in [10], hierarchical gossiping is proposed to handle one-shot evaluation of aggregate queries.

To the best of our knowledge, none of the existing systems for aggregation has been applied in the context of information retrieval, where the underlying information that needs to be aggregated refers to terms and their respective frequency values in autonomous document repositories. As a result, an open issue that remains is to what extent a distributed and scalable aggregation mechanism for term frequency values can produce retrieval results of good quality. This is one of the topics covered in this paper.

2.3. P2P Document Retrieval

Content-based search in P2P networks [25] is usually related to full-text search [14, 28, 36], with most approaches relying on the use of structured P2P networks. Some research focuses on providing P2P web search functionalities, like in [17], where MINERVA is presented, a P2P web search engine that aims at scalability and efficiency. In MINERVA, each peer decides what fraction of the web it will crawl and subsequently index. In further work, the authors also presented an information filtering approach relaxing the common hypothesis of subscribing to all information resources and allowing users to subscribe to the most relevant sources only [38].

Previous approaches regarding P2P web search have focused on building global inverted indices, as for example Odissea [27] and PlanetP [7]. In PlanetP, summaries of the peers' inverted indices are used to approximate TF-IDF. Inverse peer frequency (the number of peers containing the term) is used instead of IDF. It is questionable how this would scale in large P2P networks with dynamic contents, as also noted in [3]. In [5], super-peers are used to maintain DF for the connected peers. A similar approach is also used in [20]. Bender et al. [6] study global document frequency estimation in the context of P2P web search. The focus is on overlapping document collections, where the problem of counting duplicates is immense. Their system relies on the use of an underlying structured P2P network. A similar approach is described in [21], which is quite different from our setup that assumes an unstructured P2P architecture.

A major shortcoming of all these approaches is that their efficiency degrades with increasing query length and thus they are inappropriate for sim-

Symbol	Description
$df(t)$	Document frequency of term t
$CF(t)$	Collection frequency of term t
h	Height of DESENT hierarchy
t_j	Term
t_{size}	Size of term/frequency tuple
$tf(t,d)$	Frequency of term t in document d
N_P	Number of peers
$N_{l,i}$	Number of documents at peer P_i
P	Peer
S_Z	Number of peers in DESENT zone
T	Number of terms contributed from peer in aggregation process
TV_i	Term vector of document i

Table 1: Overview of symbols.

ilarity search. Recently, an approach has been proposed that reduces the global indexing load by indexing carefully selected term combinations [26].

The overview of an integrated system for P2P search is given in [24]. The authors propose a distributed architecture for P2P information retrieval called PHIRST. In this system the storage costs per node are reduced considerably by storing only a limited number of terms. Subsequently, a hybrid search model of both structured and unstructured search is employed at query time, for not all terms are stored within the system. However, their algorithms were tested on a test collection of moderate size and fall back to unstructured search.

2.4. P2P-based Digital Libraries (DL)

One area of research combining and applying several of the techniques introduced here are digital libraries. Several papers propose using P2P networks in a digital library context [3, 8, 9, 22, 23]. In [4], a distributed indexing technique is presented for document retrieval in digital libraries. Podnar et al. [22] use highly discriminative keys for indexing important terms and their frequencies. In [23], the authors present *iClusterDL*, for digital libraries supported by P2P technology, where peers become members of semantic overlay networks (SONs).

3. Preliminaries

Due to the resultant high number of terms found in text documents and the subsequent high dimensionality of the term vectors, the selection of a subset of terms to use for analysis and search is essential. Especially in the areas of machine learning and data mining it is a vital task to find a set of terms that both adequately represents the collection in question and filters out enough terms so that processing is computationally possible. We propose to use some of the existing techniques of filtering out less important terms on the peer level as a first filtering step before the estimation process. To this end, we first

Term selection method	Acronym	Formula
Document frequency	DF	$df(t)$
Collection frequency	CF	$\sum_{i=1}^{N} tf(t, d_i)$
Collection freq. inverse document freq.	CFIDF	$CF(t) \log \frac{N}{df(t)}$
Term frequency document frequency	TFDF	$(n_1 n_2 + c(n_1 n_2 + n_2 n_3))$

Table 2: Overview of term selection methods.

briefly provide the necessary background on the most prominent term selection techniques which can be integrated in our framework. Subsequently, we present an overview of aggregation in P2P networks, focusing mainly on a) hierarchical and b) gossip-based aggregation. An overview of symbols used throughout the remainder of the paper is given in Table 1.

3.1. Background on Term Selection

Term selection algorithms can generally be categorized as either supervised or non-supervised. Supervised methods use provided labels or class assignments for documents. The best or most discriminating terms are then selected according to their class labels and the occurrence of the term across classes (in the 20 newsgroups collection, these labels indicate the newsgroup an article originally was posted to). In many cases, however, class labels are not available. In the context of distributed collections, such labels are particularly rarely available, due to reasons of missing common document types or the general ad-hoc character of the collections themselves. To perform term selection nevertheless, unsupervised techniques – even though there exist fewer than supervised ones – can be used. These methods mainly rely on frequency information of a term or term within a collection, in order to judge its usefulness.

3.1.1. Term Selection Methods

Following the vector space model of information retrieval we use N as the number of documents in a collection (which can be either global, i.e., the whole collection, or local when only a subset of the collection is considered). Further we use *df(t)* for the number of documents a term occurs in, also called the document frequency of term t. The number of occurrences of term t in document d is denoted to as the term frequency *tf(t,d)*. In this context, we propose the usage of the unsupervised methods summarized in Table 2, as possible local term selection methods on each peer.

Document Frequency (DF). One of the most prevalent techniques is denoted as document frequency thresholding, i.e., the number of documents a certain term occurs in. The main assumptions underlying document frequency thresholding are that terms occurring in very many documents carry less discriminative information and that terms occurring only in very few documents will provide a strong reduction in dimensionality (even though they might be

discriminative in some cases). In combination with an upper and lower threshold, term selection can be applied. This generally leads to results comparable to supervised techniques.

Collection Frequency (CF). The collection frequency of a term is given by the sum of all term frequencies for a given term (the total number of occurrences of a term in a collection):

$$CF(t) = \sum_{i=1}^{N} tf(t, d_i) \tag{1}$$

Therefore, the collection frequency ranks highly terms which occur only in few documents of the collection but have a high frequency in these documents.

Collection Frequency Inverse Document Frequency (CFIDF). The CFIDF is given by weighting the collection frequency values by the inverse document frequency for a term:

$$CFIDF(t) = CF(t) \log \frac{N}{df(t)} \tag{2}$$

This measure covers both aspects; both the local document frequency and the total number of occurrences for a term.

Term Frequency Document Frequency (TFDF). Another, quite recent technique to exploit both the *tf* and *df* factors is presented in [34]:

$$TFDF(t) = (n_1 n_2 + c(n_1 n_2 + n_2 n_3)) \tag{3}$$

where n_1 denotes the number of documents in which t occurs, n_2 the number of documents t occurs only once, and n_3 the number of documents containing t at least twice. An increasing weight c gives more weight for multiple occurrences. The setting the authors recommend to use because it gave the best results in their experiments is $c = 10$. We therefore also rely on this setting.

3.1.2. Challenges and Objective

The high degree of distribution of documents in a P2P system with autonomous peers makes effective term selection particularly challenging. The reason is that when term selection is applied on a subset of the complete document collection, which resides on a peer, the terms that are deemed important may be less important when the entire collection is considered. Therefore, identifying globally important terms necessitates aggregation of local term frequency values, so that the aggregated result reflects the contents of the entire document collection. In our distributed context, it is not straightforward to choose an effective term selection method, as this is highly dependent on the degree of distribution and the representativeness of local document collections with respect to the global collection.

The main objective of this work is to identify appropriate term selection and aggregation techniques for application in large-scale P2P networks. We seek methods that produce aggregated results of comparable quality to the centralized case, where all documents would be available on a single peer.

Criterion	Hierarchical	Gossip-based
Correctness of aggregation	High	Probabilistic
Error recovery mechanism	Complex protocol	Simple protocol
Stability under stress	Unstable/prone to failure	Stable
Load balancing	Imposed on few peers	Fair load assignment
Computation cost	Reduced	Relatively high
Maintenance cost	Hierarchy maintenance	None
Communication costs	Low	High
Scalability	Aggregation at all levels	Local interactions only

Table 3: Comparison of aggregation methods.

3.2. Aggregation in P2P Networks

Aggregation is an important task for deploying useful applications in large-scale distributed systems. In the context of unstructured P2P networks, there exist two main approaches for aggregation of information: *hierarchical aggregation* and *gossip-based aggregation*.

In hierarchical aggregation, an often dynamic hierarchy is formed and information is aggregated at intermediate levels, before being propagated upwards. As a result, the aggregation process: a) is efficient due to the reduction of the communication cost, b) scalable, as the aggregation load is distributed to several peers, and c) is accurate, since the information that reaches the root accurately reflects the real aggregated value (in accordance with term ranking as shown in later experiments). On the other hand, the hierarchy induces additional maintenance cost, usually requires a complicated protocol that ensures fault-tolerance, and is unstable when the churn rate is high.

In gossip-based aggregation, peers constantly exchange information in cycles, by selecting a small random subset of other peers at each cycle. There exist theoretical guarantees that show that aggregated values converge exponentially fast to the true aggregates [11]. Gossiping is based on a simple and scalable protocol that is stable under stress because only local interactions between peers take place. Moreover, the assignment of load to all peers is fair, without any additional maintenance cost. On the downside, the guarantees for the aggregated values are probabilistic in nature and the cost for computing the aggregates is higher than in the case of hierarchical aggregation.

For comparative purposes, we provide a summary of the benefits and drawbacks of each approach in Table 3 .

4. Hybrid Aggregation of Document Frequencies

In this section, we describe our approach for aggregating terms and their document frequencies without central assembly of all data. We employ an unstructured P2P architecture and the overall aim is to provide estimates of frequency values that are as similar as possible to the centralized case, where all documents are available at a single location.

As described in the previous section, both hierarchical and gossip-based aggregation have their advantages and disadvantages. A natural question that arises is to what extent the former aggregation approaches can be combined, in order to overcome their limitations on an individual basis. This is the motivation to introduce a *hybrid* approach for aggregation of document frequency values in widely distributed unstructured P2P networks. For terms with high local frequency values, we employ hierarchical aggregation, thus computing more accurate aggregate values. Intuitively, terms with high frequency of occurrence are considered more important and have a larger impact on similarity ranking, therefore the aim is to compute their aggregate value with higher accuracy and in shorter time. The focus is then put on the remaining terms only after the dissemination of the hierarchical aggregation information. Subsequently, terms with low frequency value on local basis are aggregated using gossip-based aggregation. The aggregate frequencies of such terms will not be completely accurate, but they are approximated closely with probabilistic guarantees.

We first provide an overview of the DESENT architecture, which is used as the underlying hierarchical overlay network, and we describe how hierarchical aggregation is realized in this framework (Sec. 4.1). Then, we describe the second part of our hybrid aggregation, namely the gossip-based aggregation protocol employed (Sec. 4.2).

4.1. Hierarchical Aggregation

In order to create a hierarchical overlay network over a purely unstructured (Gnutella-like) P2P network, no matter its network distance, we employ a variant of DESENT [9]. The reasons for this choice are the completely *distributed* and *decentralized* creation of the hierarchy, its low creation cost and robustness. The most important details of the basic algorithm are described in the following; for more in-depth explanations we refer to [9]. The DESENT hierarchy can be used for building overlays for searching, but also for other purposes like aggregation of data or statistics about contents from participating peers – which is the way that DESENT is utilized in this paper.

4.1.1. DESENT

For an illustrative example of the DESENT hierarchy, see Fig. 1. The bottom level consists of the individual peers ($P_{A_1} \dots P_{A_n}$ and $P_{B_1} \dots P_{B_n}$). Then neighboring peers (network-wise) create *zones* of approximate size S_Z peers (i.e., groups of peers) around an *initiator* peer (P_A and P_B), which acts as a zone controller. Notice that the height (h) of the hierarchy equals to: $log_{S_Z} N_P$. These level 1 initiators (P_A and P_B) are mostly uniformly distributed over the network, and are selected independently of each other in a pseudo-random way. The initiators form the next level of the hierarchy, they are responsible for the peers in their zones, and they aggregate the information collected from their peers.

In the subsequent phases, super-zones are created, which consist of a number of neighboring zones from the previous level. Each super-zone is represented

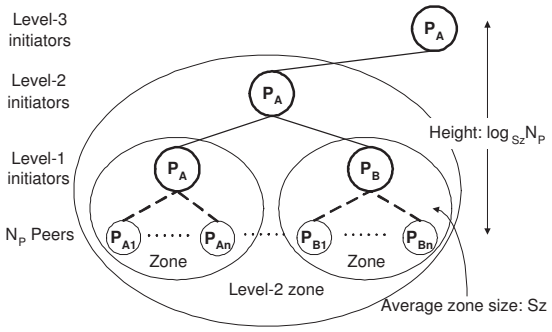


Figure 1: Example of a P2P hierarchy of height h=3 with peers and zones.

by a super-zone initiator that is responsible for the initiators in its zone and aggregates the information of these initiators. The zone initiators essentially form a P2P network similar to the original P2P network, and the aforementioned process is repeated recursively, using the zone initiators as peers. In the example of Fig. 1, P_A is initiator both at level 2 and level 3. In this way, a hierarchy of initiators is created, with each initiator collecting aggregate information that refers to the contents of all peers in the tree rooted at that initiator. Finally, at the top-level initiator, aggregate information that spans the contents of the entire network is available.

4.1.2. Aggregation Process

The process of estimating the *frequency of selected terms* based on DESENT can be summarized as follows:

1. A tree-based P2P structure is created using the DESENT protocol [8, 9].
2. All peers select up to T terms from their local document collection using one of the techniques described in Sec. 3.1, and send these terms together with the total number of documents to the parent peer in the tree.
3. Each parent peer receives up to $S_Z T$ terms with respective document frequencies, where S_Z denotes the average number of peers in a zone. The parent peer selects up to T terms, these terms are propagated upwards together with the aggregated document frequencies and the total number of documents in the subtree rooted at the peer.
4. The process continues up to the level of the children of the root (i.e., peers at level $h-1$), where h denotes the height of the tree. Level 0 is the bottom level and level h is the level of the root peer. Instead of performing the last aggregation at the root peer, it is performed by the children of the root. This is achieved by first distributing their aggregated values by hashing to the other root-children peers.
5. The estimated document frequency values and the total number of documents are disseminated to the participating peers.

6. The whole process is repeated at regular intervals, in order to capture changes in document contents, as well as improving the estimated values. An alternative to fixed-time intervals would be to employ heuristics to assess the fluctuation in the network, i.e., initiate the process once a given number of peers joins or leaves the network.

We will now describe in more detail the local term selection, document frequency calculation, aggregation, and the final dissemination of information to the peers.

4.1.3. Local Term Selection and Document Frequency Calculation

Each peer P_i selects up to T terms from the $N_{l,i}$ locally stored documents, using one of the unsupervised term selection techniques described in Sec. 3.1. Term selection at a peer is based on the peer's local knowledge only. Thus, the result of the term selection is a *term vector* TV_i, which is the number $N_{l,i}$ and vector of term tuples. Each term tuple in TV_i contains a term t_j and the local document frequency $df(t_j)$: $TV_i = [N_{l,i}, [(t_1, df(t_1)), ..., (t_T, df(t_T))]]$.

4.1.4. Level-wise Aggregation

After the $S_Z T$ selected terms from the previous phase have been received, a new term vector is created of the received terms and their frequencies, i.e., $TV_j = [N_s, [(t_1, df(t_1)), ..., (t_{S_Z T}, df(t_{S_Z T}))]]$. N_s is the sum of the received local frequencies, i.e., $N_s = \sum_{i=1}^{S_Z} N_{l,i}$. Furthermore, duplicate terms and their frequencies (i.e., the same term originating from several peers) are aggregated into one tuple. Subsequently the number of terms in the new term vector is less than $S_Z T$. Finally, the term vector is reduced to only contain T terms. Term selection is performed based on the frequency of appearance, therefore terms that have high frequency are favored. The intuition, which is also confirmed by related work in [33], is that it is important to identify terms that are globally frequent and forward such terms to the top of the hierarchy. The generated term vector after aggregation and term selection, again consisting of T terms, is sent to the next level in the tree and this process continues iteratively up to level $h-1$, i.e., the children of the root.

4.1.5. Hash-based Distribution and Aggregation

Performing the final aggregation at the root peer is a straightforward process, however it makes the system vulnerable, as it induces a single point of failure. Instead, the final aggregation is performed by the children of the root, at level $h-1$. Notice that in this phase, our approach trades efficiency for robustness. We employ a more costly way to aggregate information, however the overall system becomes fault-tolerant. The actual aggregation is achieved by having the level $h-1$ peers first distributing their aggregated values, by hashing, to the other level $h-1$ peers. A recipient peer becomes responsible for a different subset of terms and aggregates their frequencies, thus performing (part of) the task that the root peer would perform. After the aggregation of the received term vectors, the peers send all their aggregated results to the rest of the P2P

network. In the end, all level $h-1$ peers have the complete aggregated values locally available.

The reason for hashing is two-fold. First, it is important that all statistics for one particular term end up at the same node, in order to provide aggregated values per term. Second, the workload of the final aggregation is distributed and shared among the level $h-1$ peers, thus achieving load-balancing.

4.1.6. Dissemination of Information

In the final phase, the aggregated term vectors are distributed to all participating peers. This is performed by using the hierarchy as a broadcast tree. The term vectors are sent downwards, until they reach the level-0 peers. The size of the disseminated information is equal to the number of term vectors ($S_Z T$) multiplied by the number of level $h-1$ peers. The aggregated terms and document frequencies are now available at all peers locally. As a consequence, any peer can use this information, in order to provide rankings of terms and documents taking into account the global document collection. In the experimental section, we study the accuracy of relevant ranking between pairs of terms to demonstrate the effectiveness of our approach.

4.2. Gossip-based Aggregation

After having elaborated in detail how hierarchical aggregation is performed, we proceed to describe the gossip-based aggregation mechanism. Local low-frequency terms are selected for gossip-based aggregation. In practice, this selection is done by selecting the terms for which no estimation is available from the hierarchical aggregation. The rationale is twofold. First, it suffices that such aggregates are computed with probabilistic guarantees only, without the strict requirement of accurate computation. Second, aggregates of low-frequency terms can be computed with some delay, therefore gossiping can be employed leading to eventual consistency of aggregate values. In contrast, high frequency terms need to be aggregated in a more timely fashion. Thus hierarchical aggregation is a more appropriate method because they have more significant impact on search results.

4.2.1. Algorithm

The basic underlying gossiping protocol works in the following way. Any peer periodically exchanges information with another randomly selected peer. In principle, more than one peers can be selected for information exchange, however, for simplicity we assume that only one peer is selected. Each round of communication is also known as *cycle*. During a cycle, a peer exchanges its locally maintained state with that of another peer. The local state of a peer can be any value of interest that needs to be aggregated, and in the simplest case it is a plain number representing, e.g., the load of each peer.

In our setup, gossiping is used to aggregate term frequency values. Therefore, the basic intuition of gossip-based aggregation remains, only the amount of information that is exchanged between any two peers changes. Whenever two

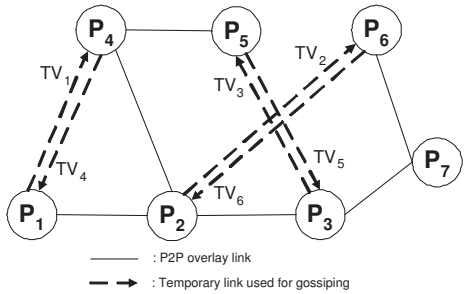


Figure 2: Gossip-based exchange of information between peers.

Algorithm 1 Gossip-based aggregation at peer P_i.

1: **while** (true) **do**
2: $P_j \leftarrow$ GetRandomPeer()
3: Send(TV_i, P_j)
4: $TV_j \leftarrow$ Receive(P_j)
5: aggregate(TV_i,TV_j)
6: **end while**

peers P_i and P_j engage in communication, their term vectors TV_i and TV_j are exchanged to perform aggregation of term frequency values. The term vectors contain information about the terms occurring on the respective peers and in how many documents they occur in following the definition given in Sec. 4. This is illustrated graphically in Fig. 2, where the peer interactions at one random gossiping cycle are shown.

Algorithm 1 provides the pseudocode for gossip-based aggregation on peer P_i. In each gossiping cycle, P_i chooses a random peer P_j (line 2) and establishes a direct link through the P2P overlay network. Then, P_i sends its term vector TV_i to P_j (line 3), and receives from P_j its term vector TV_j respectively (line 4). Then, P_i aggregates the values of the term vectors (line 5). The aggregate function can be for example the average function, so for each pair of identical terms $t_i = t_j$, their average frequency value is computed ($\frac{d_i+d_j}{2}$) and it replaces the current state (frequency) of t_i in TV_i. In the case of terms that exist on only one of the peers (e.g. P_i), they are appended to the term vector of the other peer (e.g. TV_j) and the frequency value is divided by two (e.g. $\frac{d_i+0}{2}$). Notice that at any point during the gossiping protocol, the sum of frequency values can be easily computed by multiplying the average frequency values computed thus far with the number of peers N_P in the network. An interesting aspect of the gossip-based aggregation protocol is that it can be also used to estimate the value of N_P, in case this knowledge is not available to the peers. For details on computing this type of aggregates (counting the number of peers) we refer to [11].

4.3. Combination of Hierarchical Estimation and Gossiping

The results of the hierarchical estimation is used for document frequency estimations. Terms for which there exist no hierarchical estimates (low-frequency terms) are straight-forwardly assumed to have a document frequency of one. These will subsequently be combined with the results from the gossip-based aggregation in that the document frequencies of these terms are updated to the estimated values.

5. Cost Analysis

We employ a simple cost analysis to assess the bandwidth consumption of the proposed approach. The basic parameters that influence the total communication cost (C_{total}) are: the number of peers (N_P) in the network, the average zone size (S_Z), the number of terms (T) in the term vectors propagated by each peer to its parent, the number of total terms indexed on each peer (L), the number of gossip cycles (N_C, which is usually set to $log_2 N_P$), and the size of the tuple representing each term (t_{size}). Obviously, each peer P_i can propagate T_i terms to its parent from the L_i terms indexed locally, which differ from any other peer P_j, i.e, in the general case $T_i \neq T_j$ and $L_i \neq L_j$. However, we make the simplifying assumption $T_i = T_j = T$ and $L_i = L_j = L$, in order to make the presentation of the final form of equations easier. This assumption is insinuated in the equations whenever the symbol "$\approx$" is encountered.

Each tuple of a term vector contains a term (we use as average size 16 characters for representation) and a frequency value (4 bytes). Hence, each tuple needs t_{size}=20 bytes. Moreover, each term vector is accompanied by a number (integer) that represents the number of documents associated with the term vector, however this cost is negligible compared to the size of the term vector. Notice that the height of the hierarchy (h) is derived as h=$log_{S_Z} N_P$.

5.1. Cost of Hierarchical Aggregation

The total number of terms (T_{up}) propagated upwards at all levels is calculated by multiplying the number of peers (or initiators) at that level with the number of terms (T_j) per peer P_j. Thus, the total number of terms propagated up until the children of the root are given by:

$$T_{up} = \sum_{j=1}^{N_P} T_j + \sum_{j=1}^{\frac{N_P}{S_Z}} T_j + \cdots + \sum_{j=1}^{\frac{N_P}{(S_Z)^{h-2}}} T_j \approx$$

$$N_P T + \frac{N_P}{S_Z} T + \cdots + \frac{N_P}{(S_Z)^{h-2}} T = \sum_{i=0}^{h-2} (\frac{N_P}{(S_Z)^i} T) \quad (4)$$

Thus, the cost for propagating term vectors upwards is derived as:

$$C_{up} = T_{up} t_{size} = N_P T t_{size} \sum_{i=0}^{h-2} \frac{1}{(S_Z)^i} \tag{5}$$

There exists also a communication cost (C_{out}) related to hashing the information at the children of the root. Each child hashes its $\sum_{j=1}^{S_Z} T_j \approx S_Z T$ term vectors to the other children, and the number of children is $\frac{N_P}{(S_Z)^{h-1}}$, leading to cost:

$$C_{out} = \sum_{i=1}^{\frac{N_P}{(S_Z)^{h-1}}} \sum_{j=1}^{S_Z} T_j t_{size} \approx S_Z T t_{size} \frac{N_P}{(S_Z)^{h-1}} = T t_{size} \frac{N_P}{(S_Z)^{h-2}}$$

Then all N_P peers need to recollect the aggregated term vectors, leading to a cost $C_{in} = \sum_{i=1}^{N_P} C_{out} = N_P (T t_{size} \frac{N_P}{(S_Z)^{h-2}})$ (assuming direct communication between the children of the root and the rest of the peers). Consequently, the cost for hierarchical aggregation is equal to:

$$C_{hier} = C_{up} + C_{out} + C_{in} = N_P T t_{size} (\frac{N_P + 1}{(S_Z)^{h-2}} + \sum_{i=0}^{h-2} \frac{1}{(S_Z)^i}) \tag{6}$$

Obviously, compression techniques can further reduce the total cost, however this is out of the scope of this paper. Moreover, the cost for the creation of the DESENT hierarchy is described in [9] and it is not included in this analysis.

5.2. Cost of Gossip-based Aggregation

Since each peer propagates T tuples upwards for hierarchical aggregation, we assume that each peer uses gossiping for aggregating the remaining $(L - T)$ tuples.

In each cycle, each peer P_i gossips with one randomly selected peer P_j. Thus, P_i sends to P_j a total of $(L_i - T_i)$ tuples. Therefore, the amount of information communicated in the network due to each peer in each gossiping cycle equals to: $\sum_{i=1}^{N_P} (L_i - T_i) t_{size}$. Let us further assume that N_C denotes the number of cycles employed by the gossiping protocol. Then, the total communication cost C_{gos} induced by gossip-based aggregation equals to:

$$C_{gos} = \sum_{j=1}^{N_C} \sum_{i=1}^{N_P} (L_i - T_i) t_{size} \approx (L - T) N_P N_C t_{size} \tag{7}$$

5.3. Total Cost of Aggregation

Consequently, the total cost of aggregation is straightforwardly derived and is equal to:

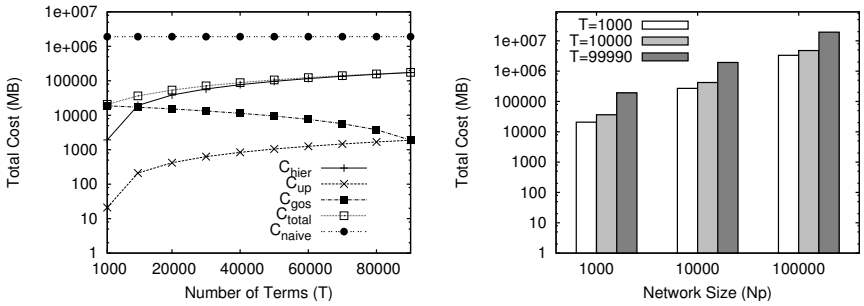


Figure 3: Cost for term aggregation and different numbers of aggregated terms and network sizes.

$$C_{total} = \underbrace{N_P T t_{size}(\frac{N_P+1}{(S_Z)^{h-2}} + \sum_{i=0}^{h-2} \frac{1}{(S_Z)^i})}_{C_{hier}} + \underbrace{(L-T)N_P N_C t_{size}}_{C_{gos}} \quad (8)$$

In Fig. 3, we provide two charts which help us derive some interesting observations, based on the cost model. Notice that the y-axis is in logarithmic scale. We set L equal to 100,000 terms, as a typical size of the vocabulary.

First, we study the aggregation cost for a network of N_P=1,000 peers and average zone size S_Z=10. We use N_C=$log_2 N_P$, as indicated by related work [11]. The total cost C_{total} equals to the hierarchical cost C_{hier} plus the cost of gossiping C_{gos}. From the chart, it seems that the dominant cost is the cost related to hierarchical aggregation, which seems counter-intuitive, since we have argued in favor of the cheap aggregation cost of hierarchical approaches. However, recall that C_{hier} also includes the cost of hash-based distribution. When we draw only the cost for pushing terms upwards (C_{up}), it becomes clear that this cost is much smaller than C_{gos}, which means that the cost of hashing is the reason of the high values of C_{hier}, and consequently C_{total}. However, the hash-based distribution is used in order to make the hierarchical aggregation more robust to bottlenecks and avoiding a single point-of-failure. Obviously, this robustness introduces an additional overhead to the total aggregation cost, and there exists an interesting trade-off between efficiency of aggregation and avoidance of bottlenecks.

In addition, in the same chart, we provide the aggregation cost C_{naive} computed as:

$$C_{naive} = N_P(N_P - 1)Lt_{size}$$

of the exhaustive approach in which each peer exchanges its complete document frequency values with all other peers. It is clear that our approach requires 1-2 orders of magnitude smaller communication cost for aggregation.

Name	#Docs	Disk	Voc. size	Avg. doc length
20 newsgroups	18.828	85M	94.753	97
DMOZ	484.113	2,9G	1.732.228	340
TREC8	528.155	2,8G	842.682	247

Table 4: Benchmark collections used in our experiments. We list the collections' names, the number of documents, the amount of disk space they use in uncompressed form, the vocabulary size of the collection, i.e., the number of distinct terms, and the average document length.

In the next chart, we graphically depict the total cost C_{total} in MB for various networks sizes (N_P) ranging from 1K to 100K peers. We use varying values for T for completeness. Notice that the total cost corresponds to approximately 20-100MB per peer, even for large network sizes. Moreover, the total cost is controlled by decreasing the T value.

6. Experimental Evaluation

All experimental code is implemented in the Java programming language. We performed experiments on a Linux server machine running 2.2 GHz Intel Xeon cpus and 18 gigabytes of memory. The virtual machine used is Sun 1.6.0_16.

We conducted experiments using three different document collections. Since we put a special focus on large-scale P2P settings, we tried to use corpora of sufficient size. In fact, two out of these three corpora contain nearly 500,000 documents, and one is smaller with about 20,000 documents. Table 4 gives an overview of the sizes of the collections used. We list the number of documents, the disk space the collections use in uncompressed form, as well as the vocabulary size of the collection, i.e., the total number of unique terms in the collections, and the average number of terms per document, i.e., the average document length. All of this information is given for the preprocessed and indexed collection.

Data collections. We worked with the 20 newsgroups data set[1] which has become very popular for text experiments in the field of machine learning and has been used for example in [18]. The data set consists of newsgroup postings from 20 newsgroups. From each newsgroup, 1,000 articles posted in the year 1993 have been selected; after removing duplicate articles (mostly cross-postings to several newsgroups), 18.828 unique messages remain.

The DMOZ collection is a collection of 483,000 web pages, which are classified by the DMOZ taxonomy[2]. The collection has been created by retrieving the web pages that are linked from the leaf-classes of the DMOZ taxonomy. The fact that the collection has been automatically retrieved from the web leads to a high

[1] `http://people.csail.mit.edu/jrennie/20Newsgroups`
[2] `http://www.dmoz.org`

number of terms in this collection. Generally, the data used here is more noisy and less focused than in the other collections, which surely has disadvantages, but definitely shows the applicability of our techniques to a general real-life web setting. The taxonomy path to a page is considered to be the class/category of the page. It is the only test collection used in this paper which is not publicly available.

Further, we used one of the collections included in the TREC information retrieval benchmarking initiative[3]. More specifically, we used the collections used in the TREC8 ad hoc evaluation. The ad hoc task is one of the traditional tasks in TREC evaluations and comprises both a collection of 500,000 to 700,000 text documents such as news messages and queries for that collection. The TREC8 collection consists of about 530,000 documents and 50 queries plus relevance judgments for them. The TREC8 collection comprises material from the Foreign Broadcast Information Service, the Los Angeles Times (randomly selected articles from 1989 & 1990), the Federal Register (1994), and Financial Times articles (1992-1994)[4].

All three test collections were preprocessed in terms of tokenization, stop word removal and stemming for the English language.

6.1. Experimental Setup

We identify the following basic parameters for our experiments and study their effect. First, the *number of partitions* or peers, as it affects the scalability of our approach. Then, the *distribution skew*, defined as the size distribution across the local partitions. A low distribution skew denotes equal amounts of documents per partition. Last, we consider the *document similarity*, defined as the degree to which documents in one partition are similar to each other. This simulates cases such as topically homogeneous collections (with a high degree of similarity) or cases of randomly distributed collections. To this end, we use class labels of documents and distribute documents to partitions already containing similar documents with a higher or lower probability according to the setting. In the case where no labels are available, document clustering is used instead to determine a measure of similarity.

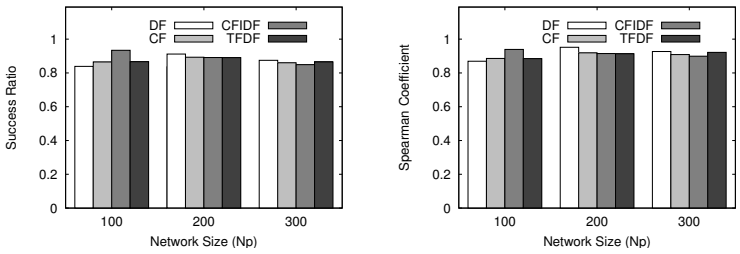
In our experimental evaluation we use varying setups, in order to simulate different use cases. We vary the number of peers to study the scalability of our approach. For each given number of peers, we apply four settings: 1) low similarity, high distribution, 2) low similarity, low distribution, 3) high similarity, high distribution, and 4) high similarity, low distribution. To be able to show the impact of all extreme values of both parameters, we also included mixed setups and also the case of documents which are distributed in equal sized partitions and have no similarity relation to each other at the other end

[3] `http://trec.nist.gov/`

[4] The TREC8 ad hoc collection consists of documents from four different collections. In this context we used the information about which document belongs to which collection only in the distribution based on similarity. There we assume documents coming from the same sub-corpus to be similar.

Id	Distribution Skew	Document Similarity within Peers
1	low	high
2	low	low
3	high	high
4	high	low

Table 5: Varying distribution skew and similarity values used in experimental setups.



(a) Success ratio. (b) Spearman coefficient.

Figure 4: Scalability with network size for the 20 newsgroups collection.

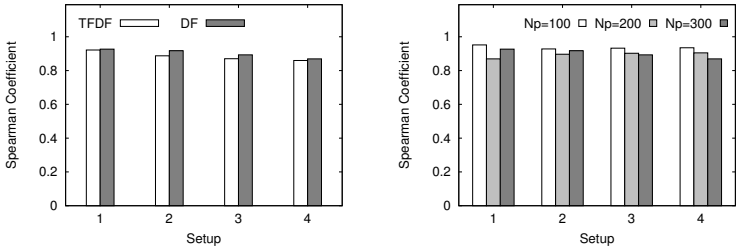
of the spectrum. We apply the aforementioned term selection methods at the local peer level to study their effect on the hybrid aggregation we propose. An overview of these four setups is given in Table 5.

6.2. Evaluation of Term Ranking Quality

Our first aim is to study the quality of the aggregated document frequencies in terms of ranking. For this purpose, we use the 20 newsgroups and the DMOZ document collections.

Evaluation metrics. We define as *success ratio* the percentage of pairs of terms that have the same relative ranking in our approach and in the centralized case. In other words, for any two terms t_i and t_j the success ratio is the fraction of the number of such pairs with the same ranking with respect to the centralized ranking, over all possible combinations of pairs of terms. We chose this performance measure for existing standard approaches such as the Spearman or Kendall tau rank order correlation coefficients lack the support for rankings of different lengths, our approach, however, is closely related and basically extends these methods in its ability to handle different lengths of involved rankings.

We additionally provide results for measuring the Spearman coefficient between the two rankings. To this end, we first need to remove elements not available in both rankings. This means that all terms for which no estimated value is available are filtered out. The resultant equally sized rankings can then be compared using the *Spearman coefficient*, a measure for correlation between two rankings, operating on the rank on their elements rather than their numeric values.



(a) Spearman coefficient for N_P=300. (b) Spearman coefficient for DF.

Figure 5: Performance for variable setups for the 20 newsgroups collection.

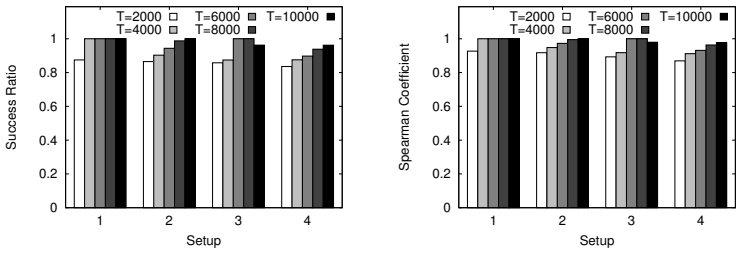
6.2.1. Results for the 20 newsgroups Collection

Fig. 4 shows the result of our scalability study with respect to the network size. For this purpose, we increase the size of the network from 100 to 300 peers. The following values are used for the experimental parameters: number of terms for hierarchical aggregation T=2,000, number of gossiping cycles N_C=20, and zone size S_Z=10 (for N_P=100) and S_Z=20 (for N_P=$\{$200,300$\}$). We use as basic setup the one with id=1, and we evaluate the following term selection methods: DF, CF, CFIDF, and TFDF.

In Fig. 4(a), we depict the values of success ratio. Regardless of term selection method, our hybrid aggregation works effectively, achieving values higher than 80% and often close to 90%. This indicates that hybrid aggregation results in high quality term rankings that are similar to the centralized case. This is verified in Fig. 4(b), when the Spearman coefficient is employed. In fact, the values of Spearman coefficient are higher than those of success ratio, again indicating the high quality aggregation of the proposed hybrid method. Another important observation is that the quality of term rankings does not deteriorate with increased network size, even when the number of peers is increased by a factor of three.

Then, in Fig. 5, we study the effect of different setups (distribution skew and document similarity within peers) on the effectiveness of our approach. In Fig. 5(a), we study the TFDF and DF term selection methods for N_P=300, and we observe that for all setups, high quality values of the Spearman metric are obtained. The best results are achieved in setup with id=1, when the distribution is low and the document similarity within peers is high. However, even for the hard setup of high distribution and low similarity, the values are still higher than 80%. This is also the case for the rest of the setups. In Fig. 5(b), we employ only the DF term selection method and we additionally vary the network size. Again the high quality values of Spearman coefficient are sustained in all setups irrespective of the size of the P2P network.

We also study the effect of increasing values of terms T that are aggregated using hierarchical aggregation in Fig. 6. Intuitively, as T increases, we expect that the quality of term ranking will improve, since more terms are aggregated hierarchically, thus resulting in more terms having accurate estimates of fre-



(a) Success ratio. (b) Spearman coefficient.

Figure 6: Effect of increasing values of T for the 20 newsgroups collection.

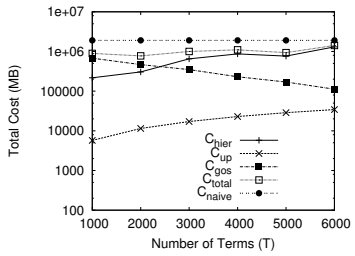


Figure 7: Experimentally derived cost for the 20 newsgroups collection and different numbers of aggregated terms.

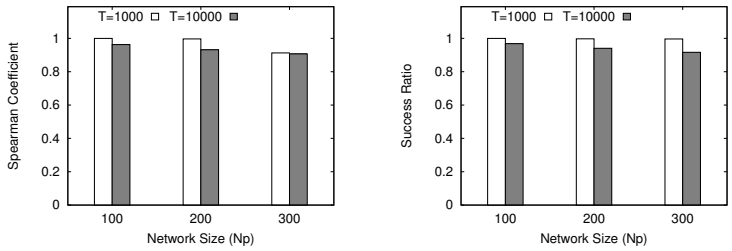
quency values. Indeed, in Fig. 6(a), we see that the values of success ratio metric increase with T irrespective of the setup. The same conclusions are drawn from Fig. 6(b), where the values of the Spearman coefficient are shown.

The cost model introduced in Section 5 is implemented as part of the experimental system. The cost we measured in experiments is depicted in Fig. 7. We show the total communication cost in MB for aggregating varying numbers of terms and the 20 newsgroups collection, Np =100. This figure is analogous to Fig. 3 which depicts the results for the theoretical performance derived from the equations of the cost model. Fig. 7 shows a high correspondence and underlines that our prototype is in accordance with the theoretical cost model.

6.2.2. Results on DMOZ Collection

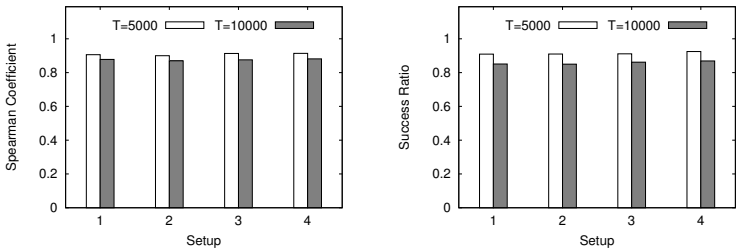
We also performed a series of experiments using the DMOZ collection, in order to study the effect of larger corpora on our hybrid aggregation method. Document collections such as DMOZ are quite challenging, because of the increased vocabulary size and the noise present in the contents.

In Fig. 8, we show the term ranking quality using the DF term selection technique. The experimental parameters in this setting are S_Z=10, N_C=10, N_P={100,200,300}, T={1,000;10,000}. In order to limit the effect of noise, we additionally use a threshold on each peer that eliminates single-occurring terms on a peer from aggregation. We observe that high quality results (always



(a) Spearman coefficient. (b) Success ratio.

Figure 8: Term ranking quality for DF for the DMOZ collection.



(a) Spearman coefficient. (b) Success ratio.

Figure 9: Term ranking quality for DF and N_P=1,000 for the DMOZ collection.

higher than 90% and often close to 100%) are obtained for both our metrics: Spearman coefficient and success ratio. When the network size increases, we observe a small decrease in the values of our quality metrics. However, the absolute values are still higher than 90%.

Another interesting observation is that increasing values of T cause a small reduction in the values of Spearman coefficient and success ratio. We believe that this is because in the DMOZ collection the increased values of T lead to many low-frequency (noisy) terms being aggregated. Even a small value of T, such as 1,000, is sufficient to aggregate the important terms. In addition, the randomness of the hierarchical aggregation makes the selection of terms (that are finally aggregated hierarchically) random as well. This is the reason that increased values of T reduce the quality of ranking.

We also performed an experiment with a large network of N_P=1,000 peers to study the scalability of hybrid aggregation. The results are depicted in Fig. 9. Again, the same conclusions are drawn verifying the performance of hybrid aggregation, even in the case of large-scale P2P networks.

6.3. Results on the TREC8 Collection

The fact that queries plus relevance judgments are available leads to several questions that can be answered in the context of P2P document frequency aggregation:

- Does the DESENT aggregation negatively influence retrieval results compared to the full information about document frequencies?
- What exactly is the tradeoff between the number of terms aggregated and efficiency in computation with regards to relevance evaluation?

We tested the retrieval performance achieved for the 50 provided queries together with the given relevance information used in the TREC evaluations. The basic question here is 'How many of the relevant documents will be retrieved?'. Each of the 50 queries is sent to the search system in order to answer this question. In the search system, we used three different settings for term weighting:

1. No document frequency information (i.e., $df = 1$ for all terms)
2. Real document frequency information obtained from the full index
3. Document frequency estimations computed by the P2P hybrid aggregation method

6.3.1. Evaluation Measures

We employ as evaluation measures recall and mean average precision. Recall is the fraction of relevant documents which are retrieved with respect to a given query:

$$Recall = \frac{\#(relevant\ items\ retrieved)}{\#(relevant\ items)} \tag{9}$$

Recall therefore measures the 'coverage' of relevant documents a search system can deliver. Precision on the other hand measures the 'conciseness' of the result, it is the fraction of retrieved documents which are relevant:

$$Precision = \frac{\#(relevant\ items\ retrieved)}{\#(retrieved\ items)} \tag{10}$$

Both recall and precision can be measured at a given number of top k items in the result set. Often one is more interested in, e.g., the precision of the results appearing on the first page of the result instead of the sometimes too high full number of retrieval results (e.g., when evaluating web search results).

Since there exist measures to incorporate both indicators in one value, we make use of such a measure called 'mean average precision (MAP)'. It is a single-figure measure for precision at all different recall levels, and has become increasingly popular within the TREC community. The MAP for a given set of queries Q is defined as:

$$MAP = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \tag{11}$$

where Q is a set of queries or information needs and m_j is the number of relevant documents for query j. R_{jk} is the set of ranked retrieval results and

$Precision(R_{jk})$ the precision at all of these levels. Then the value is averaged over the number of queries.

The mean average precision gives a more complete picture of the precision of a given query searched for in a given collection.

6.3.2. Experimental Settings

An illustrative overview of the retrieval experiments is given in Fig. 10. We first partition the collection according to the different similarity and skew distribution setups. The input for the experiments is an unstructured P2P network topology, as shown on the top of the figure. Then, we employ DESENT and gossiping for estimating the global document frequencies. Both can be applied simultaneously, since gossiping is only used for the terms not covered and aggregated upwards by DESENT. The output of this phase is the estimated document frequency values for all terms, both high-frequency and low-frequency terms. Finally, both lists are merged to provide a comprehensive estimate for as many terms as possible. We further perform retrieval experiments based on these estimated frequencies and compare to both the case of centralized (real) document ones and missing document frequency information (i.e., all document frequencies are set to one).

6.3.3. Experimental Results

The TREC8 collection has been used in many cases for relevance retrieval experiments. The reported results vary according to the additional techniques the authors used. A mean average precision of 0.3272 is, for example, reported in [13], where the authors use a combination of the probabilistic model and the language model approach. Theme-based document retrieval using an extensive lexical knowledge base is applied to the same problem in [15], and the best result reported there is a mean average precision of 0.413. Again, the authors use techniques additional to basic ranking and retrieval.

In our case, we employ a basic ranking model based on $tfidf$ weighting used in Java open source search engine Lucene[5]. All techniques we employ aim at improving document frequency estimation, which in turn could be used to improve retrieval results for different weightings. Hence, we do not compete with other groups working on this collection, basically all other ranking techniques could be used on top of our approach to improve the final results.

Tables 6, 7, and 8 show an overview of the obtained results; we list both MAP, given in Equ. 11, and recall, given in Equ. 9. We list the results on a centralized index as well as results obtained without document frequency information, available in Table 6. The experiment without *df* weighting builds the absolute baseline for further experiments. In a distributed setting it is always easily possible to assume 1 for all document frequencies.

Table 7 shows the results obtained when applying hierarchical aggregation only. It is interesting to note that the results are quite close to the centralized

[5] `http://lucene.apache.org`

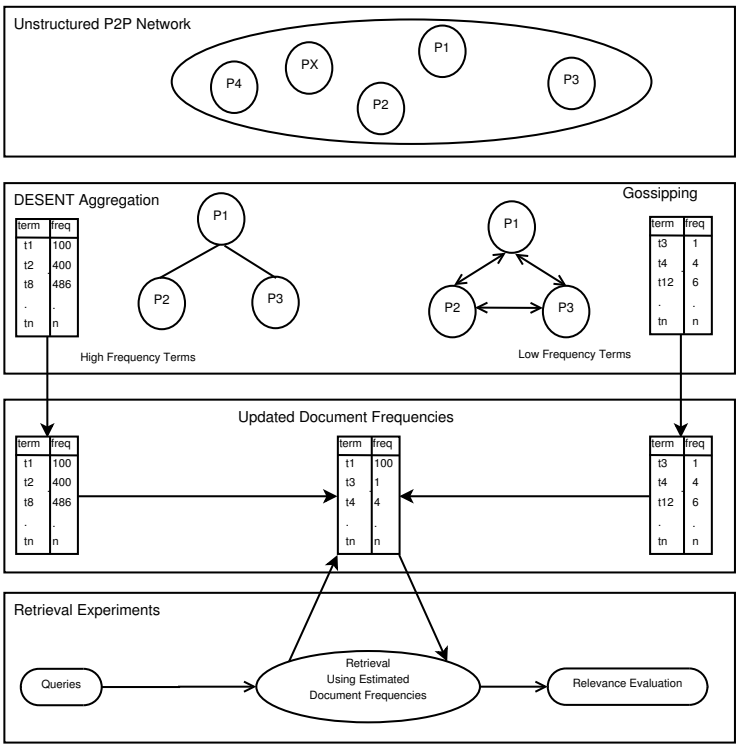


Figure 10: Overview of the architecture used in the experiments. The full system consists of DESENT, gossiping, frequency merging, and the retrieval component.

case. For instance, the average precision is around 0.215 and recall 0.631, when the corresponding values of the centralized case are 0.228 and 0.651. Moreover, the results are stable across experimental setups and the different numbers of peers apart from small fluctuations.

Furthermore, we show the results obtained when using the hybrid approach in Table 8. In this scenario, we rely on both the values obtained by hierarchical and the gossip-based aggregation. The results are always better than the hierarchical aggregation values alone consistently across all settings. For 100 peers and experimental setup id=1, the results exceed the results obtained from the centralized index. However, this is due to the limited number of queries. This approach yields better results only for 7 out of the 50 queries. Across all setups, we never get far below the results from the centralized index. In fact, for experimental setup id=1 (equal distribution and high similarity within peers), we are consistently equal to the centralized case.

We also provide an overview of estimation accuracy for document frequency values in Table 9. The results are drawn from 10 runs of retrieval evaluation and given for the four different experimental settings (see Table 5). We list the percentage of frequencies which are correctly estimated, i.e. equal to the

Setup	Mean Avg. Precision	Recall
Full centralized index (baseline)	0.228	0.651
Full centralized index without df info	0.197	0.553

Table 6: Baseline retrieval results on the TREC8 ad hoc collection. We show the retrieval results on a centralized index as well as results obtained without document frequency information available.

#Peers	Setup	Mean Avg. Precision	Recall
100	1	0.214	0.628
100	2	0.213	0.626
100	3	0.217	0.635
100	4	0.216	0.633
200	1	0.214	0.629
200	2	0.215	0.629
200	3	0.217	0.634
200	4	0.215	0.633
300	1	0.215	0.628
300	2	0.215	0.630
300	3	0.217	0.633
300	4	0.216	0.634

Table 7: Retrieval results on the TREC8 ad hoc collection. We show the retrieval results based document frequency estimation using hierarchical aggregation. Results are averaged over 10 runs.

real frequencies in the centralized case, as well as the average, minimum and maximum difference between estimated and real frequency. Further, we list the median, i.e. the difference that devides the deviations into two equal parts. A median of 11 for example means that half of the estimated frequencies differ by less than 11 from the real value. One finding is that some frequencies differ by large numbers – as seen in the maximum and mean columns. On the other hand, we see that in most of the cases the median is quite small, showing that a large part of the estimates are quite close to the real values.

Even though the estimation quality decreases with a higher number of peers, the respective retrieval experiments deliver high precision as outlined in the previous section. We therefore showed that in most cases the estimations are satisfactory, if not in the absolute value, in the form they are used in the ranking function.

7. Conclusions and Future Work

In this paper, an efficient hybrid method for aggregation of document frequency values is presented, suitable for application in loosely-coupled unstructured P2P networks. The hybrid method combines hierarchical aggregation

#Peers	Setup	Mean Avg. Precision	Recall
100	1	0.229	0.648
100	2	0.227	0.648
100	3	0.226	0.641
100	4	0.226	0.640
200	1	0.228	0.646
200	2	0.224	0.644
200	3	0.225	0.641
200	4	0.224	0.633
300	1	0.228	0.646
300	2	0.224	0.640
300	3	0.225	0.642
300	4	0.225	0.639

Table 8: Retrieval results on the TREC8 ad hoc collection. We show the retrieval results based on document frequency estimation using hybrid aggregation. Results are averaged over 10 runs of hybrid aggregation.

of carefully selected local terms with high frequency values, with gossip-based aggregation of the remaining low-frequency terms. Furthermore, we present a cost model that quantifies the communication cost of the proposed aggregation method. We also provide an extensive experimental evaluation on three document collections, assessing both term ranking quality and retrieval results, having as reference point the results obtained by a centralized system that has the complete document collection available. The results show that our hybrid aggregation performs very well for variable setups. In our future work, we intend to deploy a widely distributed information retrieval system that uses hybrid aggregation as building block for estimating document frequency values.

Acknowledgements

This work was carried out during the tenure of Christos Doulkeridis' ERCIM 'Alain Bensoussan' Fellowship Programme. Robert Neumayer is supported by grant #183337/S10 (COMIDOR) from the Norwegian Research Council.

[1] L. Azzopardi, M. Baillie, and F. Crestani. Adaptive query-based sampling for distributed IR. In *Proceedings of SIGIR'06*, 2006.

[2] M. Baillie, L. Azzopardi, and F. Crestani. Adaptive query-based sampling of distributed collections. In *Proceedings of SPIRE'06*, 2006.

[3] W.-T. Balke. Supporting information retrieval in peer-to-peer systems. In *Peer-to-Peer Systems and Applications*, 2005.

[4] 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.

#Peers	Setup	Equal DF	Max.	Mean	Median
100	1	0.14	16784.0	955.40	2.0
100	2	0.09	15516.0	1010.85	14.0
100	3	0.06	21789.0	3096.63	14.0
100	4	0.06	21882.0	2848.52	14.5
200	1	0.13	16086.0	1063.73	3.0
200	2	0.08	14977.0	1441.72	14.0
200	3	0.06	29587.0	4189.28	41.0
200	4	0.04	30887.0	4680.36	36.5
300	1	0.0	18320.0	1833.65	70.0
300	2	0.013	18806.0	1986.70	25.0
300	3	0.008	26704.0	3502.5	54.0
300	4	0.006	29450.0	4802.41	70.5

Table 9: Estimation of document frequencies on TREC8 corpus. Results are averaged over 10 aggregation runs. We show the different numbers of peers, experimental setups as well as the percentage of correctly estimated document frequencies used in TREC8 queries. We also list the maximum, mean, and median differences to the correct frequencies.

[5] W.-T. Balke, W. Nejdl, W. Siberski, and U. Thaden. Progressive distributed top-k retrieval in peer-to-peer networks. In *Proceedings of ICDE'2005*, 2005.

[6] M. Bender, S. Michel, P. Triantafillou, and G. Weikum. Global document frequency estimation in peer-to-peer web search. In *Proceedings of the 9th Int. Workshop on the web and databases*, 2006.

[7] F. Cuenca-Acuna, C. Peery, R. Martin, and T. Nguyen. PlanetP: Using gossiping to build content addressable peer-to-peer information sharing communities. In *Proceedings of HPDC'2003*, 2003.

[8] C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. Scalable semantic overlay generation for P2P-based digital libraries. In *Proceedings of ECDL'2006*, 2006.

[9] C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. DESENT: Decentralized and distributed semantic overlay generation in P2P networks. *Journal on Selected Areas in Communications*, 25(1):25–34, 2007.

[10] I. Gupta, R. van Renesse, and K. P. Birman. Scalable fault-tolerant aggregation in large process groups. In *Proceedings of DSN'2001*, 2001.

[11] M. Jelasity, A. Montresor, and Ö. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3):219–252, 2005.

[12] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proceedings of FOCS'2003*, 2003.

[13] K. L. Kwok, L. Grunfeld, and M. Chan. TREC-8 ad-hoc, query and filtering track experiments using PIRCS. In *Proceedings of TREC-8*, 1999.

[14] J. Lu and J. Callan. Full-text federated search of text-based digital libraries in peer-to-peer networks. *Information Retrieval*, 9(4):477–498, 2006.

[15] K. Mahesh, J. Kud, and P. Dixon. Oracle at trec8: A lexical approach. In *Proceedings of TREC-8*, 1999.

[16] S. Melnik, S. Raghavan, B. Yang, and H. Garcia-Molina. Building a distributed full-text index for the web. *ACM Transactions on Information Systems*, 19(3), 2001.

[17] S. Michel, P. Triantafillou, and G. Weikum. MINERVA infinity: A scalable efficient peer-to-peer search engine. In *Proceedings of Middleware'2005*, 2005.

[18] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[19] R. Neumayer, C. Doulkeridis, and K. Nørvåg. Aggregation of document frequencies in unstructured P2P networks. In *Proceedings of WISE'09*, 2009.

[20] H. Nottelmann and N. Fuhr. Comparing different architectures for query routing in peer-to-peer networks. In *Proceedings of ECIR*, 2006.

[21] O. Papapetrou, S. Michel, M. Bender, and G. Weikum. On the usage of global document occurrences in peer-to-peer information systems. In *Proceedings of COOPIS'2005*, 2005.

[22] I. Podnar, T. Luu, M. Rajman, F. Klemm, and K. Aberer. A P2P architecture for information retrieval across digital library collections. In *Proceedings of ECDL'2006*, 2006.

[23] P. Raftopoulou, E. G. M. Petrakis, C. Tryfonopoulos, and G. Weikum. Information retrieval and filtering over self-organising digital libraries. In *Proceedings of ECDL'2008*, 2008.

[24] A. Rosenfeld, C. V. Goldman, G. A. Kaminka, and S. Kraus. PHIRST: a distributed architecture for P2P information retrieval. *Information Systems*, 34(2):290–303, 2009.

[25] O. D. Sahin, F. Emekçi, D. Agrawal, and A. E. Abbadi. Contentbased similarity search over peer-to-peer systems. In *Proceedings of DBISP2P'2004*, 2004.

[26] G. Skobeltsyn, T. Luu, I. P. Zarko, M. Rajman, and K. Aberer. Query-driven indexing for scalable peer-to-peer text retrieval. In *Proceedings of Infoscale'2007*, 2007.

[27] T. Suel, C. Mathur, J. wen Wu, J. Zhang, A. Delis, Mehdi, X. L. Kharrazi, and K. Shanmugasundaram. Odissea: A peer-to-peer architecture for scalable web search and information retrieval. In *Proceedings of WebDB'2003*, 2003.

[28] C. Tang and S. Dwarkadas. Hybrid global-local indexing for efficient peer-to-peer information retrieval. In *Proceedings of NSDI'2004*, 2004.

[29] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Trans. Comput. Syst.*, 21(2):164–206, 2003.

[30] R. van Renesse and A. Bozdog. Willow: DHT, aggregation, and publish/subscribe in one protocol. In *Proceedings of IPTPS'2004*, 2004.

[31] C. L. Viles and J. C. French. Dissemination of collection wide information in a distributed information retrieval system. In *Proceedings of SIGIR'1995*, 1995.

[32] C. L. Viles and J. C. French. On the update of term weights in dynamic information retrieval systems. In *Proceedings of CIKM'1995*, 1995.

[33] H. F. Witschel. Global term weights in distributed environments. *Information Processing and Management*, 44(3):1049–1061, 2008.

[34] Y. Xu, B. Wang, J. Li, and H. Jing. An extended document frequency metric for feature selection in text categorization. In *Proceedings of AIRS*, 2008.

[35] P. Yalagandula and M. Dahlin. A scalable distributed information management system. In *Proceedings of SIGCOMM'2004*, 2004.

[36] J. Zhang and T. Suel. Efficient query evaluation on large textual collections in a peer-to-peer environment. In *Proceedings of IEEE P2P*, 2005.

[37] Z. Zhang, S. Shi, and J. Zhu. SOMO: Self-organized metadata overlay for resource management in P2P DHT. In *Proceedings of IPTPS'2003*, 2003.

[38] C. Zimmer, C. Tryfonopoulos, K. Berberich, M. Koubarakis, and G. Weikum. Approximate information filtering in peer-to-peer networks. In *Proceedings of WISE*, 2008.