Journal of Universal Computer Science, vol. 18, no. 8 (2012), 1032-1047
submitted: 23/9/11, accepted: 14/12/11, appeared: 28/4/12 © J.UCS

Automatic Tag Attachment Scheme based on Text Clustering for Efficient File Search in Unstructured Peer-to-Peer File Sharing Systems

Ting Ting Qin
(Hiroshima University, Japan
tacit@se.hiroshima-u.ac.jp)

Satoshi Fujita
(Hiroshima University, Japan
fujita@se.hiroshima-u.ac.jp)

Abstract: In this paper, the authors address the issue of automatic tag attachment to the documents distributed over a P2P network aiming at improving the efficiency of file search in such networks. The proposed scheme combines text clustering with a modified tag extraction algorithm, and is executed in a fully distributed manner. Meanwhile, the optimal cluster number can also be fixed automatically through a distance cost function. We have conducted experiments to evaluate the accuracy of the proposed scheme. The result of experiments indicates that the proposed approach is capable of making effective and efficient tag attachment in real scenarios; i.e., for more than 90% of documents, it attaches the same tags as the ones attached by human reviewers. Moreover, it proofs by the experiments that the optimal cluster number is almost the same as the number of topics from the website.

Key Words: P2P system, text clustering, automatic tag attachment, K-DMeans, TF-IDCF
Category: H.3.1, H.3.2, H.3.3

1 Introduction

With a rapid growing of Web 2.0 and the success of social network, the lack of high-quality metadata becomes a critical issue in realizing an efficient file search over the network. As a way of generating such metadata, social tagging has emerged as a good alternative to the traditional methods relying on few experts [Kim et al. 2008], and it has also drawn much attention from both academia and industry. A tag is a relevant keyword or a key phrase associated with or assigned to a certain of information (e.g., file), describing the classification of the information or the personal interest of users. In fact, in the field of Web search, users can attach tags to the resources whatever they like, and social network websites such as Flickr[1], Del.icio.us[2], CiteULike[3] and YouTube[4] have

[1] http://www.flickr.com/
[2] http://www.delicious.com/
[3] http://www.citeulike.org/
[4] http://www.youtube.com/

shown great popularity of tags and attracted millions of users. In contrast to full-text search used in crawler-based schemes, a tag-based search scheme could recommend a web page relevant to each user, by referring to tags attached to the pages by the users [Jung 2008a, Jung 2009, Jung 2011a]. And a personalization of the search result could be realized by combining the notion of tags with conventional search engines. However, users manually attach tags only to a limited number of documents, which is a laborious, time consuming work. The quality of file search in those systems could be improved through automatically attaching tags to all documents. In addition, it is a natural requirement that such a tag attachment should be done in a distributed manner [Jung et al. 2009], since documents are distributed over the network and the heavy load of centralized servers is becoming a critical issue in those systems.

Existing studies on tagging systems focus on how to improve the quality of services such as search and recommendation [Begelman et al. 2006], as well as analysing reasons of tag attachment [Golder and Huberman 2006, Jung 2008b, Jung 2010a, Jung 2012a]. Nevertheless, considering the potential usefulness of the notion of tags, automatic tag attachment is a research topic which has received less attention than it deserves. In addition, free and voluntary tagging adopted by most of web search engines can suffer from ambiguity and arbitrariness. The problem comes from the fact that several users usually generate different tags for the same document and even a single user's tagging behavior may change over time. Therefore finding methods to realize automatic tag attachment is becoming an increasingly important topic [Song et al. 2011].

For such reasons, an automatic tag attachment scheme [Qin and Fujita 2011] was proposed firstly by the authors to improve the quality of file search in P2P environment. It is a combination of text clustering with a modified tag extraction algorithm. In this paper, the original algorithm is continually optimised by the authors. More concretely, in the improved scheme, the first step is to divide the whole documents space into several clusters by exploiting a distributed K-DMeans algorithm whereas the optimal number of clusters could be fixed automatically through a distance cost function. Such a constraint function is significant to decrease the clustering cost as well as algorithm complexity. Then the second step is to extract tags from each cluster by adopting a modified TF-IDF. It is noteworthy that several tags automatically generated by the scheme may not appear in the given document, and could refine the classification of the documents to improve the quality of file search. Although the proposed cluster-based scheme is applied only on text documents at the current stage, such an idea can be extended to other type of documents.

The remainder of this paper is organized as follows. Sect. 2 overviews related work. Sect. 3 outlines a model of Peer-to-Peer (P2P) system [Qin et al. 2009] which realizes a quick file search based on the notion of tags. Sect. 4] describes the proposed tag attachment scheme. The result of preliminary experiments is given in Sect. 5. Finally, Sect. 6 concludes the paper with future works.

2 Related Work

The objective of text clustering is to partition a given set of documents into clusters of similar documents. Such a clustering is particularly effective to improve the quality and the efficiency of information retrieval in distributed systems [Papapetrou et al. 2010]. Conventional text clustering schemes can be classified into three types, i.e., partitioning type such as K-Means [Liu et al. 2010], hierarchical type such as BIRCH [Zhang et al. 1996], and their hybrid such as Scatter/Gather [Ke et al. 2009]. Although there are several distributed schemes proposed in the literature [Dhillon and Modha 1999, Forman and Zhang 2000], most of them assume the existence of a centralized server which controls the overall process of the text clustering.

During the growth of the semantic web [Kim et al. 2010], the domain of social tagging has gained increasing attention which has resulted in significant outcomes. Thus, the application of semantics to knowledge management [Garcia-Crespo et al. 2009][Garcia-Crespo et al. 2010] becomes more and more important [Davies et al. 2007]. Meanwhile, social tagging in P2P environments has also attracted considerable attention in recent years due to the widespread adoption of P2P network. Tagster [Görlitz et al. 2008] is an open source P2P tagging system. It adopts a data management scheme called PINTS, which is based on the Vector Space Model (VSM) and a distributed data structure similar to Distributed Hash Table (DHT). PINTS maintains a feature vector for each user, and by using such vectors, it predicts possible respondents, and utilizes the search results returned by those potential respondents to indicate candidate tags. More concretely, it keeps a distributed index structure to hold user-tag, user-resource, and resource-tag relations, and when a user requests a resource with a given ID, it returns all tags attached to the resource with information about all users who attached tags to it.

Recently, a system called P2PDocTagger [Ang et al. 2010] was proposed to support automatic tag attachments. It learns how documents are tagged by the users based on a classification method, and shares such knowledge among peers to realize a tag attachment to other documents. Another personalized tag recommendation system called PUT-Tag was built considering the tag, user and content similarity together in recommending tags [Hamouda and Wanas 2011]. However, it relies on a manual tag attachment by the users, which is still an expensive and time consuming task.

3 Tag-Based P2P File Search System

3.1 Hierarchical P2P Architecture

This section describes an outline of P2P system which realizes a quick file search using the notion of tags [Qin et al. 2009]. The tag attachment scheme proposed in the current paper assumes the existence of such a file search system, although the evaluation conducted in [Section 5] does not rely on a specific P2P architecture.

Consider a three-tier P2P architecture consisting of top, middle, and bottom layers. The top layer consists of centralized server, the middle layer consists of a number of sub-servers, and the bottom layer consists of a large number of user peers (UPs, for short). In the following, the authors denote the central server by C, and a set of sub-servers as $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$.

In this system, the central server C takes the responsibility of keeping the correlation between files held by the UPs and sub-servers, and forwarding queries received from UPs to sub-servers relevant to the query, where in practice, C could be realized as a collection of peers to avoid possible bottlenecks. Each sub-server S_i in the middle layer is associated with a group of UPs, and takes the responsibility of storing file indexes uploaded by the associated UPs, and processing queries received from C or a UP contained in the corresponding group. Several UPs in the bottom layer can be grouped according to the similarity of users' interests behind the peers, and/or the proximity of their geographical locations.

In the following, some details of a file search algorithm proposed in the previous paper [Qin et al. 2009] are described. Firstly, two basic tools will be introduced in [Sections 3.2.1 and 3.2.2]. Secondly, a way of associating each UP to a group corresponding to a sub-server as well as a way of collecting file indexes held by UPs to the corresponding sub-servers are described in [Section 3.2.3]. Afterwards, a way of quickly identifying sub-servers relevant to a given query is described in [Section 3.2.4]. At last, some detail problems in these two processes will be discussed in [Section 3.2.5].

3.2 File Search Algorithm

3.2.1 Tag-Based Sieving of Files

In this system, the central server C maintains a set of tags $T = \{t_1, t_2, ..., t_n\}$, which plays two roles in the proposed system. On the one hand, it is used as a kind of index in the process of a file search. On the other hand, it is used to define a group of UPs and to associate a sub-server to a given group of UPs, i.e., in this sense, tags contained in T must be a low-frequency but a representative word in some sense.

3.2.2 Priority Sequence of Tags

Let σ be a bijection from T to $\{1, 2, \ldots, |T|\}$. In the following, $\sigma(t)$ is referred to as the **priority** of tag t, and it is said that tag t_1 is given a higher priority than tag t_2 under σ, if $\sigma(t_1) < \sigma(t_2)$. Note that function σ naturally defines the following sequence of tags, which will be referred to as a **priority sequence** of tags, in what follows:

$$\sigma^{-1}(1), \sigma^{-1}(2), \ldots, \sigma^{-1}(|T|),$$

where σ^{-1} denotes an inverse of function σ.

Next, the notion of "inclusion relation" between tag sets is introduced, which plays an important role in the proposed file search scheme.

Definition 1: Let $T_1, T_2 \subseteq T$ be two subsets of tags. T_1 is said to be included by T_2 under σ, denoted by $T_1 \sqsubseteq_\sigma T_2$, if the priority sequence of T_2 is a prefix of the priority sequence of T_1, where we assume that any T_1 is included by the empty set.

Example: Let $T = \{t_1, t_2, \ldots, t_9\}$ and assume that $\sigma(t_i) < \sigma(t_{i+1})$ for $1 \leq i \leq 8$. Subset $T_1 = \{t_1, t_2, t_3\}$ is included by subset $T_2 = \{t_1, t_2\}$ under σ, since the priority sequence of T_2, i.e., t_1, t_2, is a prefix of the priority sequence of T_1, which is t_1, t_2, t_3. On the other hand, subset $T_3 = \{t_2, t_3, t_4\}$ is not included by $T_2 = \{t_1, t_2\}$ under σ, since the priority sequence of T_2, i.e., t_1, t_2, is not a prefix of the priority sequence of T_1, which is t_2, t_3, t_4.

Definition 2: Two tag sets T_1 and T_2 ($\subseteq T$) are said to be incomparable under σ, if neither $T_1 \sqsubseteq_\sigma T_2$ nor $T_2 \sqsubseteq_\sigma T_1$.

A function to check the inclusion of T_1 by T_2 is described as follows:

```
function INCLUSION($T_1, T_2$)

Step 1: If $|T_1| < |T_2|$, then return false and stop, where $|T|$ denotes the cardi-
    nality of set $T$.

Step 2: If $T_2 = \emptyset$, then return true and stop.

Step 3: Let $t_1$ be a highest priority tag in $T_1$, and $t_2$ be a highest priority tag
    in $T_2$. Let $T_1 := T_1 \setminus \{t_1\}$ and $T_2 := T_2 \setminus \{t_2\}$.

Step 4: If $t_1 \neq t_2$, then return false and stop. Otherwise, go to Step 2.
```

3.2.3 File Uploading Process

This subsection describes a way of uploading indexes of files held by a UP, to a particular sub-server. Each sub-server is associated with a subset of tags, and each file held by a UP is attached at least one tag by the user. The algorithm associates files with sub-servers through the notion of inclusion of tag sets. A concrete procedure, which is executed by each UP holding indexes to be uploaded, is described below.

procedure FILE_UPLOAD

Step 1: Let $\hat{T}$ be the set of tags attached to the file index to be uploaded.

Step 2: Find a sub-server S_i which is associated to a tag set T^* such that $\mathsf{INCLUSION}(\hat{T}, T^*)$ is true.

Step 3: Connect to sub-server S_i and upload the file index to S_i.

This procedure is invoked by a UP when a file is newly created and/or the contents of a file are modified by the UP. A request of uploading indexes is handled by the central server C to determine a sub-server to which the given file index should be transferred.

3.2.4 Query Forwarding Process

Next, the process of query forwarding is considered, which is the key operation in the search process. The main difference between this three-tier P2P search engine and conventional search engines is that the central server plays a role of controller to balance the network traffic in the whole system. A system variable NR, indicating the total number of files discovered so far, plays a similar role to the TTL in flooding-based schemes; i.e., every time a new file is discovered, NR is incremented by one, and the search process stops when NR reaches a predefined value. A pseudo-code for the query forwarding process is given below.

procedure QUERY_FORWARD

Step 1: Let $\tilde{T}$ be the set of tags corresponding to a query q received by the central server C.

Step 2: C identifies a sub-server S_i associated to a tag set T^* such that $\mathsf{INCLUSION}(\tilde{T}, T^*)$ is true.

Step 3: C connects to S_i and forwards q to S_i.

Step 4: After receiving q, S_i conducts a file search similar to conventional search engine, and directly notifies the result to the requesting UP. The number of matching results is notified to C.

Step 5: If the number of matching results is smaller than predetermined NR, C tries to find another sub-server S_j such that the associated tag set T_j contains at least one of the same tags in $\tilde{T}$, and go to step 3. Otherwise, it stops.

3.2.5 Discrimination Tree

The key point in the processes of file uploading and query forwarding is that how to confirm a sub-server responsible for the above mentioned situations. To solve this problem, in this scheme, each sub-server is associated with a subset of tags in such a way that for any subset T' of T, there exists a sub-server which is associated with a set of tags including T' under σ concerning with **inclusion** function. Such an assignment of tags can be represented by a tree structure, namely **discrimination tree**, described below:

- Each vertex in the tree is associated with a set of tags. In the following, let $T(u)$ denote a set of tags associated to vertex u in the tree.
- The root of the tree is associated with an empty set of tags.
- Let u be a vertex in the tree, and t' be the lowest priority tag in $T(u)$. Let $i' = \sigma^{-1}(t')$ for brevity. Then, in the tree structure, vertex u has no children or it has $n - i'$ children associated with a tag set $T(u) \cup \{\sigma(j)\}$ for each $i' + 1 \leq j \leq n$.
- Each leaf in the tree corresponds to a tag set associated with a sub-server, and a sub-server associated to a leaf plays the role of its parent if it is the left-most child of the parent (such assignment of the role of parent is recursively conducted until it reaches the root vertex).

Observe that a collection of resultant sets of tags certainly satisfies the requirement described above. such tree structure is also used for the "discrimination" of a given query, in a sense that a query received from a client is placed at the root vertex, and moves toward a leaf vertex associated with a tag set including the query. The time required for determining a sub-server relevant to a given query is proportional to the depth of a leaf vertex relevant to the query.

It is noteworthy that a combination of tagging with P2P file search system overcomes typical limitations of traditional P2P systems such as resource allocation. Therefore, tag plays an important role on improving the efficiency of file search in P2P environment. In next section, the authors will introduce an approach of how to attach tags to files automatically.

4 Proposed Scheme

4.1 Overview

In this paper, an automatic tag attachment scheme is proposed for documents distributed over a P2P network. The authors assume that the underlying P2P system consists of sub-servers and user peers (UPs). UP owns documents, generates vectors and file indexes from its documents, and uploads them to its corresponding sub-server. Sub-server serves as a cluster manager; i.e., it generates a cluster index, extracts tags from uploaded information, and maintains them by periodically communicating with its corresponding UPs.

The proposed scheme proceeds in two steps; i.e., after conducting a text clustering with the notion of modified K-DMeans method, it applies a TF-IDCF algorithm to automatically extract representative tags for each cluster. In the following, after describing the basic clustering framework in [Section 4.2], the details of the modified K-DMeans method is described in [Section 4.3]. Finally, a way of automatic tag extraction for each cluster is shown in [Section 4.4].

4.2 Distributed Clustering Framework

In the proposed scheme, k sub-servers are selected firstly from the given set of peers according to the computational power and the communication bandwidth. Let $\mathcal{P}$ be the set of sub-servers. A subset of vectors will be (dynamically) associated with each sub-server, by assuming that the sub-server can collect all information associated with it. Each sub-server in $\mathcal{P}$ executes the following algorithm collaboratively.

```
Algorithm DISTRIBUTED_CLUSTERING

Step 1: Let $D_i$ be a part of given data set $D$ associated with sub-server $P_i \in \mathcal{P}$.
    Note that the union of $D_i$'s equals to $D$. $P_i$ "calculates" the center $x_i^*$ of
    subset $D_i$, and disseminates it to all sub-servers.

Step 2: After receiving centers of other subsets, $P_i$ partitions $D_i$ into $k$ subsets
    $D_{i1}, D_{i2}, \ldots, D_{ik}$ according to the distance to $x_j^*$, and sends $D_{ij}$ to $P_j$ for
    all $j \neq i$.

Step 3: After receiving subsets $D_{1i}, D_{2i}, \ldots, D_{ki}$, $P_i$ updates $D_i$ as
    $D_i := \bigcup_{j=1}^{k} D_{ji}$.

Step 4: If it satisfies the termination condition, then terminate the algorithm.
    Otherwise, go to Step 1.
```

In the above framework, it needs to determine the following points to actually calculate a clustering of D: 1) how to calculate the initial partition of D; 2) how to calculate the center of D_i; and 3) how to determine the termination condition of the algorithm. In the next subsection, each point will be described in detail.

4.3 Modified K-DMeans Method

The refinement of a clustering in each iteration proceeds as follows. Let $C = \{c_1, c_2, \ldots, c_k\}$ be a variable representing the set of centers. Initially, elements in C are randomly selected from the vector space of data set, and at the end of the execution, the algorithm terminates if C does not change after the update in an iteration. The key idea of the proposed method is that during the t^{th} iteration, several data which have the highest similarity with the center of the $(t-1)^{th}$ cluster will be selected, then a mean of them will be used as the center of the t^{th} clustering. More concretely, the calculation of the "next" centers proceeds as follows.

procedure Next_Centers

Step 1: For each D_i, let Δ_i be the lowest similarity between c_i and a document in D_i, where the similarity between two items is calculated by the cosine similarity in a VSM. Note that Δ_i corresponds to a radius of cluster D_i.

Step 2: Calculate a core D'_i of cluster D_i, which is a collection of documents whose similarity is no less than $1-\delta(1-\Delta_i)$, where δ is a parameter satisfying $0 \leq \delta \leq 1$. Then, calculate the "mean" of D'_i and regard it as the center of D_i.

The reader should note that in Step 2, D'_i is not empty since it contains c_i even if δ is arbitrarily small. On the other hand, it contains all documents in D_i if $\delta = 1$; i.e., the cardinality of D'_i can be controlled by setting an appropriate value to parameter δ. Although the modified K-DMeans method can find a partition of data for a fixed number of clusters, one target of a cluster validity procedure is to set automatically the optimal number of clusters. However, it can not be clearly and easily confirmed in fact due to its uncertainty. The authors recommend a distance cost function to confirm the optimal cluster number. At first, the conditions of optimal solution k_{opt} and its upper bound k_{max} are presented.

Definition 1: Let D_{outer} be the sum of distance between each c_i and the center of the whole data set w.

$$D_{outer} = \sum_{i=1}^{k} || \, c_i - w \, || \tag{1}$$

Definition 2: Let D_{inner} be the sum of inner distance of each cluster. Here the inner distance for a cluster is the sum of distance between each data x in this cluster and c_i.

$$D_{inner} = \sum_{i=1}^{k} \sum_{x \in c_i} || \, x - c_i \, || \tag{2}$$

Definition 3: Let $F(c,k)$ be the validity function.

$$F(c,k) = D_{outer} + D_{inner} \tag{3}$$

Therefore, the problem of finding the optimal number of cluster is equal to find the k that minimize the value of $F(c,k)$;i.e.,

$$k_{opt} = min_k\{F(c,k)\} \tag{4}$$

Then the definition of the average cluster outer distance is given as well as the average cluster inner distance as follows:

$$\bar{d}_{outer} = \frac{D_{outer}}{k} \qquad \bar{d}_{inner} = \frac{D_{inner}}{n} \tag{5}$$

where n is the total number of documents.

Considering that when a document space has the characteristic of fractal geometry; i.e., the space structure of each cluster is similar to the whole document space. Then it satisfies

$$\frac{\bar{d}_{inner}}{D_{inner}/k} = \frac{\bar{d}_{outer}}{D_{outer}} \tag{6}$$

While when a document space does not have the fractal geometry characteristic, a well performed clustering must guarantee that the distance among each cluster center should be as large as possible, well the inner distance between each data and the cluster center should be as small as possible. Then it satisfies

$$\frac{\bar{d}_{inner}}{D_{inner}/k} < \frac{\bar{d}_{outer}}{D_{outer}} \tag{7}$$

Using these equations it can be deduced that $k \leq \sqrt{n}$; i.e., the upper bound k_{max} is equal to$\sqrt{n}$. It is significant that the upper bound can be used to decrease the clustering cost.

Next, the cost for the clustering is evaluated as follows. Firstly, it spends a certain time on selection of centers for each cluster, especially on computing the similarity between document vectors, of which the time complexity is referred to $O\left(n' \log n'\right)$, where n' is the number of vectors in a cluster. Secondly, considering that the number of clusters $k \leq \sqrt{n}$, therefore the total time complexity of clustering approaches to $O\left(n' \times \sqrt{n} \times \log n'\right)$. This means that the cost grows logarithmically, thus it can scale to network with large numbers of documents and clusters.

4.4 Automatic Tag Extraction of Each Cluster

TF-IDF weight is a well-known metric to represent the significance of a term in a document, which combines document-specific local characteristics called term frequency (TF) and collection-specific global characteristics called inverse document frequency (IDF). Unfortunately, in fully distributed systems such as P2P, the cost to acquire a global characteristics such IDF is very expensive so that it should be better to use an alternative way such as random sampling and/or approximation. In addition, TF-IDF gives a score for each document independently, and does not explicitly take into account the similarity of documents contained in the same cluster. Typically, it is emphasised here that tags may not necessarily be contained within the documents. They are chosen from words which are close related to, or best reflect the document. Therefore, it is natural to assume that two documents contained in the same cluster are attached similar tags, regardless of containment of specific terms in the document (e.g., a document should be attached term "baseball" if it belongs to the same cluster with articles concerned with baseball even if it does not contain term "baseball").

To overcome such problems, a new metric called TF-IDCF was developed, which is a modification of TF-IDF such that it calculates IDF in each cluster. More concretely, TF-IDCF weight $W(t,d)$ of term t in document d is calculated as follows:

$$W\left(t,d\right) = \frac{\mathsf{tf}(t,d) \times \log\left(\frac{D_i}{n_{it}} + 0.01\right)}{\sqrt{\sum_{t \in d}\left(\mathsf{tf}\left(t,d\right) \times \log\left(\frac{D_i}{n_{it}} + 0.01\right)\right)^2}} \tag{8}$$

where tf denotes the term frequency and n_{it} is the number of documents in D_i containing term t. With the notion of TF-IDCF, given set of document vectors corresponding to a cluster, automatic tag attachment proceeds as follows: (a) remove several stop words using ordinary text mining technique, (b) select several terms with high TF-IDCF weight as tags representing the cluster (in the following experiments, the number of terms is fixed to five), and (c) attach extracted tags to all documents in the cluster.

5 Experiments

5.1 Setup

The authors evaluate the performance of the proposed scheme by experiments. Programs used in the experiments were written in Java under the following environment: open-SUSE/10.1, Eclipse Helios/3.6.1, and JDK/1.6. As a benchmark, a set of documents drawn from 20 topics in CiteULike is chosen. Those documents are mapped to papers indexed in CiteSeer[5] each of which is manually

[5] https://citeseer.ist.psu.edu/

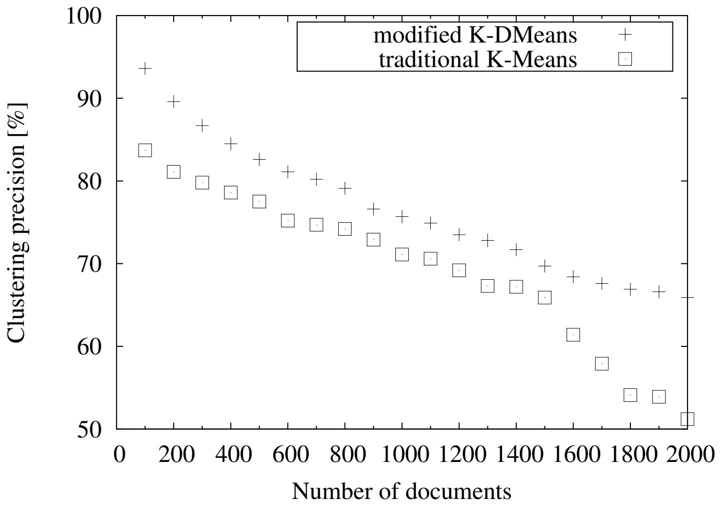

Figure 1: Clustering precision.

attached tags by the users, where the total number of distinct tags is 2017 and each document is attached 4.6 tags on average.

The authors used PeerSim 1.0.4 [Jelasity et al. 2009] to simulate the behavior of the underlying P2P. The number of peers is fixed to 100 and except for the last experiment, the number of leaders is fixed to 20 so that it is equal to the number of topics. Although it is not concretely described here, it is certified that the execution time of the proposed scheme is almost the same as a centralized scheme in which the number of leaders is fixed to one. In fact, the amount of data exchanged among leaders is sufficiently small compared with the whole network traffic.

Finally, parameter δ used in the proposed scheme is fixed to 0.8 according to the result of preliminary experiments. Recall that δ is a parameter used to refine the number of core documents in each iteration; i.e., by decreasing δ from 1.0, it could effectively eliminate the influence of documents which are far from the current cluster center, while if it is too small, it could not avoid a situation in which there are no documents other than the cluster center in the resulting core.

5.2 Results

At first, the precision of clustering is evaluated in terms of the percentage of the number of documents which are correctly classified into 20 given topics,

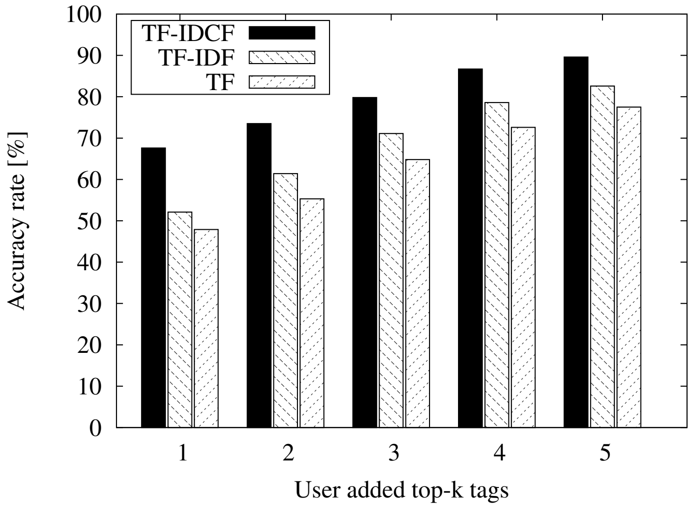

Figure 2: Tagging precision (a).

by varying the size of data set D from 100 to 2000. [Fig. 1] shows the result, where the horizontal axis is the size of data set D. The precision of the proposed method is higher than the original method, although it decreases as increasing the size of D. It is apparently due to the effect of parameter δ; i.e., by explicitly extracting a core of each cluster, the center of the next cluster can be determined more accurately than the previous scheme.

Next, the authors evaluate the precision of the set of tags extracted by TF, TF-IDF and TF-IDCF, and examine whether the top-x tags have a non-empty intersection with the set of tags manually attached by the users by varying x from one to five. The result is shown in [Fig. 2]. For $x = 1$, the precision of TF-IDCF is 67.6%, whereas that of TF and TF-IDF are 47.9% and 52.1%, respectively. As the value of x increases, the precision gradually increases, and more importantly, the precision of TF-IDCF significantly outperforms other schemes. Especially when $x = 5$, the precision of TF-IDCF reaches 89.6%, whereas that of TF and TF-IDF are 77.5% and 82.6%, respectively.

At last, the authors evaluate the tagging precision by varying the number of clusters from 15 to 30. [Fig. 3] shows the result. The tagging precision increases as increasing the number of clusters from 15 to 25, which is because documents in each cluster are strongly related with each other, as well as the difference among clusters become more evident. However, it degrades when the number of clusters becomes 30, whereas the relation among documents is broken up due to the increase of the number of clusters. Therefore, it can be concluded that the

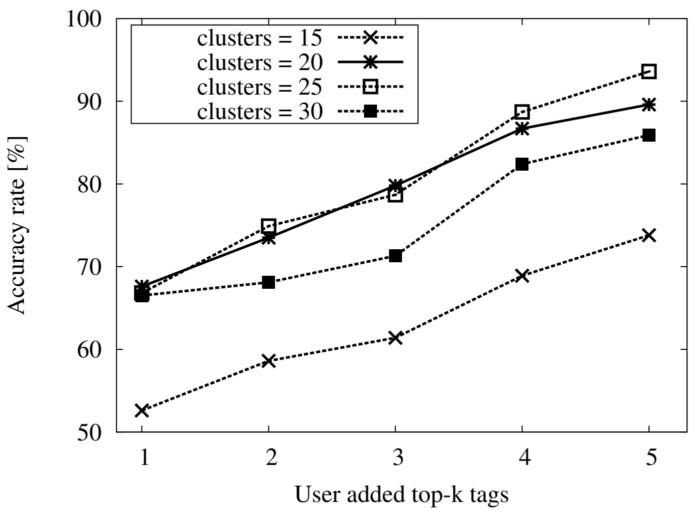

Figure 3: Tagging precision (b).

topics from the website is almost the optimal clusters for current data.

6 Concluding Remarks

This paper presented an improved scheme for automatic tag attachment to documents distributed over a P2P system. The results of experiments indicate that the proposed scheme improves the original scheme based on K-Means method and TF-IDF, as well as guarantees the adaptability of tag attachment to practical systems. The future work will be as follows: First, the accuracy of the clustering should be enhanced by refining the definition of similarity of items. The way of applying this scheme to the P2P file search system is another important issue. The next step will be focused on how to improve the tagging efficiency, as well as a large-scale experiment with large-quantity data collection in P2P environment.

References

[Ang et al. 2010] Ang, H. H., Gopalkrishnan, V., Ng, W. K., Hoi, S. C. H.: "P2PDocTagger: Content management through automated P2P collaborative tagging"; Proc. 36th Intern. Conf. VLDB, (Sep 2010).

[Begelman et al. 2006] Begelman, G., Keller, P., Smadja, F.: "Automated tag clustering: Improving search and exploration in the tag space"; Proc. Collaborative Web Tagging Workshop in conjunction with 15th Intern. Conf. WWW'06, (May 2006).

[Dhillon and Modha 1999] Dhillon, I. S., Modha, D. S.: "A data-clustering algorithm on distributed memory multiprocessors"; Proc. Large-Scale Parallel KDD Systems Workshop in conjunction with ACM SIGKDD Intern. Conf. KDD, (Aug 1999).

[Davies et al. 2007] Davies, J., Lytras, M. D., Sheth, A. P.: "Semantic-web-based knowledge management"; IEEE Internet Computing, 11, 5 (2007), 14-16.

[Forman and Zhang 2000] Forman, G., Zhang, B.: "Distributed data clustering can be efficient and exact"; ACM SIGKDD Explorations Newsletter, 2, 2, ACM, New York (Dec 2000), 34-38.

[Golder and Huberman 2006] Golder, S. A., Huberman, B. A.: "Usage patterns of collaborative tagging systems"; Journal of Information Science, 32, 2 (Apr 2006).

[Garcia-Crespo et al. 2009] Garcia-Crespo, A., Colomo-Palacios, R., Gomez-Berbis, J. M., Mencke, M.: "BMR: Benchmarking Metrics Recommender for Personnel issues in Software Development Projects"; International Journal of Computational Intelligence Systems, 2, 3 (2009), 256-266.

[Garcia-Crespo et al. 2010] Garcia-Crespo, A., Colomo-Palacios, R., Gomez-Berbis, J. M., Garcia-Sanchez, F.: "SOLAR: Social link advanced recommendation system"; Future Generation Computer Systems, 26, 3 (2010), 374-380.

[Görlitz et al. 2008] Görlitz, O., Sizov, S., Staab, S.: "Tagster- tagging-based distributed content sharing"; Proc. 5^{th} Euro. Conf. ESWC'08, (Jun 2008), 807-811.

[Hamouda and Wanas 2011] Hamouda, S., Wanas, N.: "PUT-Tag: personalized user-centric tag recommendation for social bookmarking systems"; Social Network Analysis and Mining, 1, 4 (2011), 377-385.

[Jelasity et al. 2009] Jelasity, M., Montresor, A., Jesi, G. P., Voulgaris, S.: "The PeerSim simulator"; Expert Systems with Applications, 36, 7 (2009), 10627-10633.

[Jung 2008a] Jung, J.J.: "Ontology-based Context Synchronization for Ad Hoc Social Collaborations"; Knowledge-Based Systems, 21, 7 (2008), 573-580.

[Jung 2008b] Jung, J.J.: "Query transformation based on semantic centrality in semantic social network"; Journal of Universal Computer Science, 14, 7 (2008), 1031-1047.

[Jung 2009] Jung, J.J.: "Knowledge Distribution via Shared Context between Blog-based Knowledge Management Systems: a Case Study of Collaborative Tagging"; Expert Systems with Applications, 36, 7 (2009), 10627-10633.

[Jung et al. 2009] Jung, J.J., Lee, H., Choi, K.S.: "Contextualized Recommendation Based on Reality Mining from Mobile Subscribers"; Cybernetics and Systems, 40, 2 (2009), 160-175.

[Jung 2010a] Jung, J.J.: "An Evolutionary Approach to Query Sampling for Heterogeneous Systems"; Expert Systems with Applications, 37, 1 (2010), 226-232.

[Jung 2010b] Jung, J.J.: "Reusing Ontology Mappings for Query Segmentation and Routing in Semantic Peer-to-Peer Environment"; Information Sciences, 180, 17, (2010), 3248-3257.

[Jung 2011a] Jung, J.J.: "Ubiquitous Conference Management System for Mobile Recommendation Services Based on Mobilizing Social Networks: a Case Study of u-Conference"; Expert Systems with Applications, 38, 10 (2011), 12786-12790.

[Jung 2011b] Jung, J.J.: "Service Chain-based Business Alliance Formation in Service-oriented Architecture"; Expert Systems with Applications, 38, 3 (2011), 2206-2211.

[Jung 2011c] Jung, J.J.: "Boosting Social Collaborations Based on Contextual Synchronization: An Empirical Study"; Expert Systems with Applications, 38, 5, (2011), 4809-4815.

[Jung 2012a] Jung, J.J.: "Attribute selection-based recommendation framework for short-head user group: An empirical study by MovieLens and IMDB"; Expert Systems with Applications, 39, 4 (2012), 4049-4054.

[Jung 2012b] Jung, J.J.: "Discovering Community of Lingual Practice for Matching Multilingual Tags from Folksonomies"; Computer Journal, 55, 3 (2012), 337-346.

[Jung 2012c] Jung, J.J.: "Online Named Entity Recognition Method for Microtexts in Social Networking Services: a Case Study of Twitter"; Expert Systems with Applications, 39, 9 (2012), 8066-8070.

[Kim et al. 2008] Kim, C., Jung, J. Y., Zin, H. C., Jung, J.J.: "An application of meta search agent system based on semantized tags for enhanced web searching"; Journal of Universal Computer Science, 14, 14 (2008), 2400-2415.

[Ke et al. 2009] Ke, W., Sugimoto, C. R., Mostafa, J.: "Dynamicity vs. effectiveness: Studying online clustering for scatter/gather"; Proc. 32^{nd} Intern. Conf. ACM SIGIR Research and Development in Information Retrieval, ACM, New York (Jul 2009).

[Kim et al. 2010] Kim, H., Ji, A., Ha, I., Jo, G.: "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation"; Electron Commer Res Appl, 9, 1 (2010), 73-83.

[Liu et al. 2010] Liu, Y., Xiao, S., Lv, X., Shi, S.: "Research on K-means text clustering algorithm based on semantic"; Proc. 10^{th} Intern. Conf. CCIE'10, 1 (Jun 2010), 124-127.

[Papapetrou et al. 2010] Papapetrou, O., Siberski, W., Fuhr, N.: "Text clustering for Peer-to-Peer networks with probabilistic guarantees"; Proc. 32^{nd} Euro. Conf. ECIR'10, (2010).

[Qin and Fujita 2011] Qin, T., Fujita, S.: "Automatic Tag Attachment Scheme for Efficient File Search in Peer-To-Peer File Sharing Systems"; Proc. Advances in Social Network Analysis and Mining (ASONAM 2011), (Jul 2011), 507-511.

[Qin et al. 2009] Qin, T., Cao, Q., Wei, Q., Fujita, S.: "A Hierarchical Architecture for Real-Time Search in Peer-To-Peer Networks"; Proc. PDAA Workshop in conjunction with Intern. Conf. PDCAT'09, (Dec 2009), 482-487.

[Song et al. 2011] Song, Y., Zhang, L., Giles, C.: "Automatic tag recommendation algorithms for social recommender systems"; ACM Trans Web (TWEB), 5, 1 (Feb 2011), 1-31.

[Zhang et al. 1996] Zhang, T., Ramakrishnan, R., Livny, M.: "BIRCH: An efficient data clustering method for very large databases"; Proc. ACM SIGMOD Intern. Conf. Management of Data, 25, 2, ACM, New York (Jun 1996).