PINTS: Peer-to-Peer Infrastructure for Tagging Systems

Olaf Görlitz, Sergej Sizov, Steffen Staab
University of Koblenz-Landau, Germany
Dept. of Computer Science
{goerlitz, sizov, staab}@uni-koblenz.de

ABSTRACT

Self-organizing structure and availability of almost unlimited resource capacities make the peer-to-peer architecture very attractive for large-scale sharing of annotated data in Web 2.0 scenarios. This paper addresses the problem of information aggregation and utilization in a decentralized tagging environment. We introduce the vector space model for characterization of users, resources, and tags. We analyze the problem of constructing a reliable approximation for feature vectors in a fully decentralized setting and introduce possible solutions. The results of large-scale systematic evaluation with realistic data sets witness the viability of our approach.

1. INTRODUCTION

The rapidly growing popularity and data volume of modern Web 2.0 tagging applications originate in their ease of operating for even unexperienced users, suitable mechanisms for supporting collaboration, and attractivity of shared annotated material (images in Flickr, videos in YouTube, bookmarks in del.icio.us, etc.). However, existing centralized (and mainly commercial) systems have a number of serious limitations, including limited resource allocation (e.g. users being charged for content beyond the strict limitation of free space), complete service unavailability due to maintenance or denial-of-service attacks, and the need to trust multiple independent services if different resource types shall be shared.

Decentralized and self-organizing infrastructures plus huge amounts of available resources make modern peer-to-peer (P2P) systems highly attractive for applications with collaborative annotation (tagging) of multimedia contents. This reason has motivated active recent research efforts in the field of peer-to-peer systems for Web 2.0 environments. A good recent example of a P2P-based tagging system is the open source project *Tagster* [1], which allows for decentralized annotation and distributed sharing of locally stored resources (such as user photos, bookmarks, videos, and the like) without central coordinating instances or servers at all.

Characterization of available peers, tags, and resources by IR-like feature vectors (e.g., based on frequencies of their occurrences and co-occurrences in the system) may help to construct intelligent algorithms for ranked retrieval. Suggesting tags for annotating new postings, visualizing so-called tag clouds, predicting user's favorite resources, or recommending to join relevant groups of interest are just few examples.

[1] available at `http://isweb.uni-koblenz.de/Research/tagster`

At first glance, it appears natural to adopt the established information retrieval (IR) methodology. Common IR methods usually combine document-specific local characteristics (e.g., term frequency $tf(t)$ of term t in document d) and collection-specific global characteristics (e.g., inverse document frequency $idf(t) = logN/N_t$ for term t, document collection of cardinality N, and its fraction of size N_t of documents that contain t) in order to assign meaningful weights to particular dimensions of the feature vector.

Unfortunately, reliable estimation of the feature vector becomes non-trivial in a decentralized setting. Of course, each peer still knows its local statistics (i.e. *tf* values) and can directly use them or share them with others. However, global framework statistics (i.e. *idf* values) cannot be obtained directly at zero cost.

In presence of distributed index structures (e.g., DHT-based Chord index for P2P network maintenance and key lookups), the value of N can be estimated by periodically running in a decentralized manner a suitable counting routine (e.g., [10]). At the same time, values N_t can be obtained from index peers that are responsible for the key t and maintain a list of all 'relevant' peers that offer some contents associated with t. We can however expect that the dynamic nature of a collaborative tagging system will cause frequent updates of index lists and thus continuously changing values N_t. Hereby *idf* values associated with different keys t may change over time at very different rates. It is clear that comprehensive expert tuning of heuristically chosen thresholds or custom update intervals to address this problem would cause prohibitively high administration overhead.

The PINTS algorithm aims to overcome stated problems by considering the history of particular *idf* values and predicting their evolution in the future. The quality of peer's representation is driven by a (dis)similarity threshold between true and approximated versions of its feature vector. Consequently, our algorithm asks responsible index peers for notification about new *idf* values only beyond critical deviations from the estimated pattern (i.e. considerable deviations that would lead to the consistency violation). As a result, the number of required updates of *idf* values is drastically reduced.

The rest of this paper is organized as follows. In Section 2 we formalize our notion of the vectors space model for tagging environments. Section 3 introduces the PINTS algorithm for maintaining reliable estimates of peer's feature vectors in a decentralized environment. Section 4 explains our evaluation methodology and shows results of systematic evaluation on realistic large-scale data sets obtained from the

tagging platforms Flickr and del.icio.us. Section 5 discusses related work. Section 6 concludes and shows directions of our future work.

2. THE VECTOR SPACE MODEL

The structure of a collaborative tagging framework can be seen as a tripartite network [8] with ternary relations (tag assignments) between users $u \in U$, resources (e.g. images, media files) $r \in R$ and associated tags (arbitrary text labels, in our case) $t \in T$. The set of all relations of the tagging framework is therefore $Y \subseteq U \times T \times R$ [11]. These relations can be used for characterization of network elements in a variety of ways. For instance, the user can be characterized by used tags and/or published resources; particular tags can be characterized by users (which used them) and/or resources (that the tags annotate). A number of applications, including personalized recommender systems, tag suggestions for annotating new resources, or mining of special interest groups, can be built on top of the corresponding representational model, as we show in our related work [12] that is conducted in parallel, with respect to the methodology presented in this paper. To generalize various forms of mutual element characterization in the tagging system, we establish the generic notion of so-called *tag clouds* and cloud-specific feature vectors, and then discuss ways of efficiently approximating them in a decentralized setting.

2.1 Tag clouds

The natural way to characterize elements from U and R in a tagging framework is a collection of element-specific tags $t_i \in T, i = 1..k$.

DEFINITION 2.1. *A tag cloud is defined as the tuple*

$$\mathcal{T} := (Y^*, f) \tag{1}$$

where $Y^* \subseteq Y$ is a context-dependent subset of all tag relations and

$$f(t) : Y^* \rightarrow \mathbb{R} \tag{2}$$

is a tag-rank function that computes a score/value for each $t \in T^* \subseteq Y^*$.

The introduced notion of tag clouds can be directly used to characterize elements from U and R. For example, the *user-centric* tag cloud $\mathcal{T}_u$ (i.e. user-specific collection of tags for the user $u \in U$) and a *resource-centric* tag cloud $\mathcal{T}_r$ (i.e. all associated tags of a single resource $r \in R$) can be expressed as

$$\mathcal{T}_u := (Y_u, f_u), \quad Y_u \subseteq u \times T \times R, \tag{3}$$
$$\mathcal{T}_r := (Y_r, f_r), \quad Y_u \subseteq U \times T \times r, \tag{4}$$

Analogously, more complex *community-centric* tag clouds can be used to summarize all tags used by a group of users $U^* \subseteq U$ or for a collection of resources $R^* \subseteq R$:

$$\mathcal{T}_{U^*} := (Y_{U^*}, f_{U^*}), \quad Y_{U^*} \subseteq U^* \times T \times R \tag{5}$$
$$\mathcal{T}_{R^*} := (Y_{R^*}, f_{R^*}), \quad Y_{R^*} \subseteq U \times T \times R^* \tag{6}$$

Finally, a combination of (5) and (6) results in a tag cloud of a community around users and resources

$$\mathcal{T}_{U^*R^*} := (Y_{U^*R^*}, f_{U^*R^*}), \quad Y_{U^*R^*} \subseteq U^* \times T \times R^*, \tag{7}$$

2.2 Constructing Feature Vectors

Due to the tripartite nature of Y^* the distinct item sets (i.e. U, T, R) are mutually characterizing each other. Therefore, we define the domain d of an item set I to be the set of the respective other two item sets: $d(I) = (J, K)$. Hence $d(U) = (T, R)$, $d(T) = (U, R)$, and $d(R) = (T, U)$.

Generic Weighting.

We now present a generic model to characterize any kind of item set by computing weights for the elements in their domain. The combination of term frequency and inverse documents frequency $tf \cdot idf$ is commonly used in information retrieval for weighting terms of text documents. Following the similar motivation, we introduce the notion of *item-to-item frequency* and an *inverse item frequency*.

DEFINITION 2.2. *The item-to-item frequency is defined as*

$$if(i) = (a_i, b_i) \tag{8}$$

with a_i and b_i being the number of occurrences of i in a relation with either of the other item sets in $d(I)$.

DEFINITION 2.3. *The inverse item frequency is defined as the ratio between cardinalities of $d(I)$ and of its subset appropriate as permutation of elements that have a tag relation with i:*

$$iif(i) = \left(\log \frac{|J|}{|J^*|}, \log \frac{|K|}{|K^*|}\right), \; j \in J^* / k \in K^* : (i, j, k) \in Y^* \tag{9}$$

DEFINITION 2.4. *The overall weight(i) of the i^{th} element in the feature vector is defined as a vector product $if(i) \cdot iif^T(i)$.*

Feature vectors for characterization of resources (using relationships to associated users and tags) or tags (using relationships to associated resources and users) can be constructed in an analogous manner.

Example: Figure 1 shows a small-sized tag cloud consisting of three users, three resources (e.g. images), three tags, and the respective tag assignment (red, green, blue lines). For the sake of clarity, we consider in this example only tag-centric features for user characterization and remove irrelevant relationships (edges) between users and resources.

Figure 1: Cloud example: tag relations

To construct the tag-based feature vector of the user *alice* we use the whole tag cloud $\mathcal{T}$ for calculating $iif(i)$ and the subset $\mathcal{T}_{alice}$ (c.f. (3)) for calculating the user-centric $if(i)$:

$$weight(holiday) = (1,1) \cdot (log\frac{3}{1}, log\frac{3}{1})^T = 2 \cdot log3 \simeq 3.2$$
$$weight(mountain) = (1,1) \cdot (log\frac{3}{2}, log\frac{3}{2})^T = 2 \cdot log\frac{3}{2} \simeq 1.2$$
$$weight(sea) = (1,1) \cdot (log\frac{3}{2}, log\frac{3}{1})^T \simeq 2.2$$

We notice that more 'characteristic' features (e.g. tag 'holiday' that was used by user *alice* once to annotate only one resource) get higher values than less 'discriminative' ones (e.g. tag 'mountain' is shared by two users and annotates two resources). The resulting feature vectors of the three users in our example are as follows:

$$\begin{aligned} v_{alice} &= (2 \cdot log3,\ 2 \cdot log\tfrac{3}{2},\ log3 + log\tfrac{3}{2}) \simeq (3.2,\ 1.2,\ 2.2) \\ v_{bob} &= (0,\ 2 \cdot log\tfrac{3}{2},\ 0) \simeq (0.0,\ 1.2,\ 0.0) \\ v_{dave} &= (0,\ 0,\ log\tfrac{3}{2} + log3) \simeq (0.0,\ 0.0,\ 2.2) \end{aligned}$$

For computing similarity between feature vectors v_1 and v_2 we use the common notion of IR-style cosine measure:

$$sim(v_1, v_2) = \frac{v_1 \cdot v_2^T}{||v_1|| \cdot ||v_2||} \tag{10}$$

We notice that users *alice* and *bob* from our sample scenario use the same tag *mountain* but do not have any shared resources. The second pair *alice* and *dave* has the tag *sea* in common and shares the resource *photo3*. These observations are reflected by similarities (10) between corresponding feature vectors:

$$\begin{aligned} sim(v_{alice}, v_{bob}) &= 0.29 \\ sim(v_{alice}, v_{dave}) &= 0.54 \end{aligned}$$

To allow for more flexible tuning of feature vectors, we extend Definition 2.4 by arbitrary weighting coefficients as follows:

$$if^{\alpha}(i) = \alpha^T \cdot if(i) \tag{11}$$

$$\alpha = (\alpha_1, \alpha_2), |\alpha| = 1 \tag{12}$$

$$weight^{\alpha}(i) = if^{\alpha}(i) \cdot iif^T(i) \tag{13}$$

For instance, by defining for our example ($\alpha_1 = 0,\ \alpha_2 = 1$) or ($\alpha_1 = 1,\ \alpha_2 = 0$), we obtain alternate feature vectors for the user *alice* as follows:

$$\alpha_1 = 0, \alpha_2 = 1 \ : \ v^{\alpha}_{alice} = (log3, log\frac{3}{2}, log3) \tag{14}$$

$$\alpha_1 = 1, \alpha_2 = 0 \ : \ v^{\alpha}_{alice} = (log3, log\frac{3}{2}, log\frac{3}{2}) \tag{15}$$

Example (14-15) shows that our general framework can be used for constructing various application-specific feature spaces for tunable characterization of users, tags, or resources. In the following sections, we will primarily consider the baseline scenario of user characterization by associated tags. However, the same methodology can be used for other possible feature spaces and application scenarios without any limitations.

3. THE PINTS APPROACH

This section formalizes the problem of consistent representation for peer feature vectors in a decentralized tagging environment. Consequently, we substantiate our PINTS approach for continuously maintaining high-quality approximations of the feature vectors at low communication overhead.

3.1 Updates in a decentralized environment

According to Definition 2.4, the $weight(i)$ of each component i in the feature vector is composed using two parts, $if(i)$ and $iif(i)$. According to the definition 9, values of $iif(i)$ are global statistics that depend on all postings captured by the corresponding tag cloud and thus not directly available to its particular peers. Each particular peer u needs to estimate them by communicating with index peers of the framework which maintain a DHT-based distributed index and are responsible for collecting information about peers associated with particular resources $r \in R$ and tags $t \in T$, and answering corresponding key lookups in the network. We assume that index peers can compute corresponding values $iif(i)$ exactly. Particular peers $u \in U$ do not have transient access to this information and use some locally maintained approximation $iif^*(i)$ which leads to a certain error of their locally maintained estimate $v^*_u(\theta)$. The goal is to ensure that the similarity (cf. (10)) between true and estimated feature vectors of u remains reasonably high, i.e.

$$sim(v_u, v^*_u) > \delta \tag{16}$$

with custom similarity threshold δ (e.g. $\delta = 0.9$). The consistent approximation $iif^*(i)$ can be maintained using different update strategies:

Transient propagation. The naive solution would be to propagate updates after each new posting immediately to all peers that are interested in knowing $iif(i)$, i.e., all peers that offer contents associated with i. If the number of such peers is large, immediate multicast of each minor update would however cause extremely high message complexity.

Fixed intervals. Updating $iif^*(i)$ to most recent value $iif(i)$ periodically with fixed intervals (e.g. daily or weekly) helps to reduce the required network traffic. Between two updates, the estimate $iif^*(i)$ on peer u remains constant. As long as update periods are properly tuned, this strategy may provide fairly reasonable results. However, particular values $iif(i_1)$ and $iif(i_2)$ for different elements i_1 and i_2 may change at fully different rates. It is clear that comprehensive customization of update intervals would cause prohibitively high administration overhead.

PINTS approximation. Our approach aims to predict the evolution of $iif(i)$ using a reasonably small number of k values $iif(i, \theta_1) \ldots iif(i, \theta_k)$ at timepoints $\theta_1..\theta_k$ from the past. In other words, we represent the evolution of $iif(i)$ by the time series function with time factor θ: $iif^*(i, \theta)$ that provides the best fit for observations $iif^1(i)..iif^k(i)$. Due to the almost linear evolution of data points within short time frames PINTS uses the approximation $iif^*(i, \theta) = a_i \cdot \theta + b_i$ with custom coefficients a_i and b_i that are estimated e.g. by linear regression. The entire feature vector v_u is represented by the time series function $v^*_u(\theta)$.

The update strategy of PINTS can be summarized as follows. Each peer u maintains locally the approximation $v^*_u(\theta)$. Under the assumption that the evolution model for all features $j \neq i$ is perfect, a compressed form of this function $v^*_{u,i}(\theta)$ is submitted to the index peer that is responsible for item i. When information about new postings change the iif value on the index peer to iif_i^{NEW}, it verifies the consistency condition (16). If this condition is violated, the index peer notifies the corresponding peer u by submitting the new approximation $iif^*(i, \theta)$ of iif's further evolution; in turn, the peer provides to the index peer its updated function $v^*_u(\theta)$ for further monitoring $iif(i)$.

3.2 Example

As an example, we instantiate the PINTS approach for the user characterization scenario already discussed before.

In compliance with our notion introduced in Section 2, we assume that each user $u \in U$ acts as a peer in the overlay network which forms at the same time his "global" tag cloud $\Im_u$ (3). For the sake of clarity, we concentrate on the user characterization by tags (i.e. using $\alpha_1 = 0$ and $\alpha_2 = 1$ in our previously introduced notion (13)).

Let $N(\theta) = |R(\theta)|$ and $N_t(\theta) = |R_t(\theta)|$, where $R(\theta)$ denotes the set of resources in the network at timepoint θ and $R_t(\theta) \subset R(\theta)$ denotes its subset that is annotated by tag t. Additionally, let $iif(t,\theta) = log(N(\theta)/N_t(\theta))$ and $if_u(t,\theta)$ denotes the number of resources that the user u annotated with tag t. In this case $v_u^*(t,\theta) = if_u(t,\theta) \cdot iif^*(t,\theta)$. For each tag t, the peer u uses the approximation $iif^*(t,\theta) \simeq a_t \cdot \theta + b_t$ in order to construct the feature vector $v_u^*(\theta)$. These approximations are managed by corresponding index peers that are responsible for keys t in the overlay network.

Now we assume that the index peer responsible for a certain tag t_m receives at some timepoint θ information about new postings and updates the corresponding iif value to $iif_{t_m}^{NEW}$. The feature vector v_u^* of the peer u and the feature vector v_{u,t_m}^* of the index peer (responsible for t_m) are now as follows:

$$v_u^*(\theta) = \begin{pmatrix} if(t_1)\cdot(a_{t_1}\cdot\theta + b_{t_1}) \\ \vdots \\ if(t_m)\cdot(a_{t_m}\cdot\theta + b_{t_m}) \\ \vdots \\ if(t_N)\cdot(a_{t_N}\cdot\theta + b_{t_N}) \end{pmatrix} \quad v_{u,t_m}^*(\theta) = \begin{pmatrix} if(t_1)\cdot(a_{t_1}\cdot\theta + b_{t_1}) \\ \vdots \\ if(t_m)\cdot iif_{t_m}^{NEW} \\ \vdots \\ if(t_N)\cdot(a_{t_N}\cdot\theta + b_{t_N}) \end{pmatrix}$$

By rearranging the cosine similarity (10) between these vectors around powers of θ and $iif_{t_m}^{NEW}$, we obtain the consistency condition (16) in the following form:

$$\frac{v_u^*(\theta)\cdot v_{u,t_m}^*(\theta)}{||v_u^*(\theta)||\cdot||v_{u,t_m}^*(\theta)||} > \delta \quad \leftrightarrow \quad \frac{X}{Y\cdot Z} > \delta$$

$$X = A_{t_m}\theta^2 + 2B_{t_m}\theta + C_{t_m} + if(t_m)^2\cdot iif_{t_m}^{NEW}\cdot(a_{t_m}\theta + b_{t_m})$$

$$Y = \sqrt{A_{t_m}\theta^2 + 2B_{t_m}\theta + C_{t_m} + if(t_m)^2\cdot(a_{t_m}\theta + b_{t_m})^2}$$

$$Z = \sqrt{A_{t_m}\theta^2 + 2B_{t_m}\theta + C_{t_m} + if(t_m)^2\cdot(iif_{t_m}^{NEW})^2}$$

with A_{t_m}, B_{t_m}, and C_{t_m} defined by

$$A_{t_m} = \sum_{t_i \neq t_m} if(t_i)^2 a_i^2, \quad B_{t_m} = \sum_{t_i \neq t_m} if(t_i)^2 a_i b_i, \quad C_{t_m} = \sum_{t_i \neq t_m} if(t_i)^2 b_i^2 \tag{17}$$

With each posting, an updated set of A_{t_m}, B_{t_m}, and C_{t_m} values is sent to the index peer of tag t_m which itself knows $if(t_m)$, $iif_{t_m}^{NEW}$ and the previously calculated a_{t_m} and b_{t_m}. Thus, the index peer has all the information necessary to check the consistency condition (16) and to decide whether a notification to peer u is necessary.

The storage requirement per tag t_m and user u at index peers for the data structure [*userid*, if_{t_m}, a, b, A, B, C] is 56 *bytes* if 8 *bytes* are used per element. Additional space is required for the history of *iif* values per tag t_m, e.g. 80 *bytes* with a history length of five values and $2\cdot 8$ *bytes* for the value/timestamp tuple. Considering the Flickr dataset with 320,000 users and 1.6 million tags and an average of 50 users per tag, we get an overall space requirement of 4,6 GB. With an even distribution of tags across 1000 index peers (c.f. section 4.2) each peer would have to store on average a mere index size of 4,6 MB.

To illustrate the introduced method, we extend the previously discussed application scenario by assuming $N = 3$ and the following history of resource-related values N_t for $t \in \{holiday,\ mountain,\ sea\}$ with corresponding *iif* values and approximated coefficients a_t, b_t for $iif_t^*(\theta)$:

t	$N_t(\theta{=}0)$	$N_t(\theta{=}1)$	$iif_t(\theta{=}0)$	$iif_t(\theta{=}1)$	a_t	b_t
holiday	0	1	0	$log3$	$log3$	0
mountain	2	2	$log\frac{3}{2}$	$log\frac{3}{2}$	0	$log\frac{3}{2}$
sea	1	1	$log3$	$log3$	0	$log3$

The dynamic approximation of the feature vector (14) has now the form $v_{alice}^*(\theta) = (log3\cdot\theta, log\frac{3}{2}, log3)$.

At the timepoint $\theta{=}2$, the index peer responsible for $t = holiday$ receives information about new postings and realizes that the *iif* value has changed to $iif_{holiday}^{NEW} = 3\cdot log3$. The similarity

$$sim(v_{alice}^*(\theta{=}2), v_{alice,holiday}^*(\theta{=}2)) =$$
$$sim((2\cdot log3, log\frac{3}{2}, log3), (3\cdot log3, log\frac{3}{2}, log3)) = 0,9890$$

satisfies the consistency condition if δ is set to 0.9, so that an immediate notification to *alice* about the changed $iif_{holiday}^{NEW}$ is not necessary.

4. EVALUATION

4.1 Reference Data Sets

Our large-scale reference datasets were obtained by systematically crawling the Flickr and Del.icio.us portals during 2006 and 2007. The target of the crawling activity were the core elements, namely users, tags, resources and tag assignments[2]. The statistics of the crawled datasets are summarized in Table 1.

	users	tags	resources	tag assignm.
flickr	319,686	1,607,879	28,153,045	112,900,000
del.icio.us	532,924	2,481,698	17,262,480	140,126,586

Table 1: Flickr and Del.icio.us dataset statistics

For crawling the Flickr dataset we applied the following strategy. First, we started a tag centric crawl of all photos that were uploaded between January 2004 and December 2005 and that were still present in Flickr as of June 2007. For this purpose, we initialized a list of known tags with the tag assignments of a random set of photos uploaded in 2004 and 2005. After that, for every known tag we started crawling all photos uploaded between January 2004 and December 2005 and further updated the list of known tags. We stopped the process after we reached the end of the list.

For the Del.icio.us crawl, the most recent postings were monitored over a period of several months to collect an initial list of user names. Afterwards, all user pages were crawled for corresponding postings and newly discovered users were added to the list. For all the users contained in the data set, we collected the almost complete information about their postings as of December 2006.

Both datasets show a continuous growth in number of users, tags and resources. There are also certain points in time with significant increase of the growth rate as both systems became more popular.

[2]The reference data sets used for this evaluation are available at `http://isweb.uni-koblenz.de/Research/DataSets`

4.2 Experimental Design

We simulated various application scenarios for the introduced model in order to evaluate its correctness and feasibility. To ensure the objectivity of experiments (an important point is that the evaluation framework should not have been designed with same model behavior in mind), we used the third-party open source PeerSim modeling framework [1] that was released fully independently and earlier than the methodology of this proposal. Each user of the tagging framework was modeled as an individual peer. The number of *index* peers (i.e. online peers that maintain the DHT based index) was set to 1000 throughout all experiments. Even with peers joining and leaving the network this minimum number can always be ensured for maintaining the decentralized index. Increasing the number of index peers only results in a marginal larger message complexity for the interval and PINTS approach while all similarity measures remain unchanged.

4.3 Results

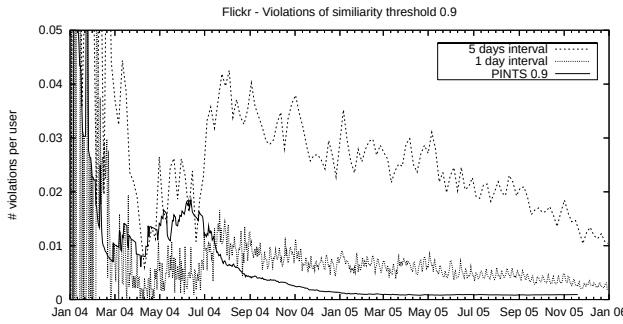
In our evaluation, we systematically compared previously discussed methods of maintaining approximations for feature vectors of peers: immediate propagation of all changes, periodic updates with constant intervals, and our PINTS method. Main metrics of interest were the message complexity (i.e., the number of update messages transmitted in the network) and the corresponding accuracy of approximation (reflected by the relative frequency of similarity threshold violations). These statistics were obtained for different intervals (constant-interval updates) and different similarity thresholds (PINTS).

Similarity threshold violations. To verify the quality of approximation, we checked the ability of all algorithms to maintain the reasonable similarity of predicted feature vectors to their real values. For interval updates we measured the number of violations at the similarity threshold $\delta = 0.9$, which was also given to the PINTS algorithm as a tuning parameter. In figures 2(a) and 3(a) we show the average relative frequency of threshold violations per user (i.e., the total count of violations, normalized by the number of users existing in the system at the corresponding timepoint). In addition, the immediate update propagation has obviously always a similarity of 1.0 (to this reason, it is not explicitly shown in figures 2(a) and 3(a)).

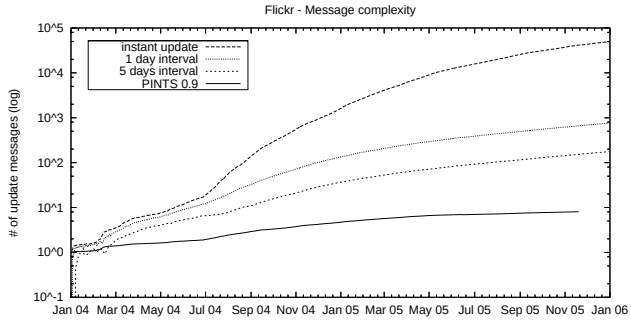
The fluctuations of the interval curves can be explained by different tagging intensity within particular intervals. In general, longer intervals between updates lead to worser approximation of feature vectors and to more violations of the similarity threshold.

Message complexity. In figures 2(b) and 3(b) we show the number of update messages that have been transmitted between peers and index peers in order to propagate changed iif values. As expected, the immediate (instant) propagation of all updates causes the highest message complexity. In all experiments, we observe the natural growth of absolute values due to the rapidly increasing number of tags, users and postings (i.e. the growth of the tagging framework itself). To this reason, all values in these charts are normalized by the total number of tags in the framework, and logarithmically scaled.

The interval curves clearly show that small update intervals result in higher message complexity. On the other hand, we observe that the PINTS message complexity (e.g. with



(a) Relative frequency of violations of the similarity threshold $\delta = 0.9$ with different update strategies



(b) Message complexity for different update strategies

Figure 2: Results for the Flickr data set

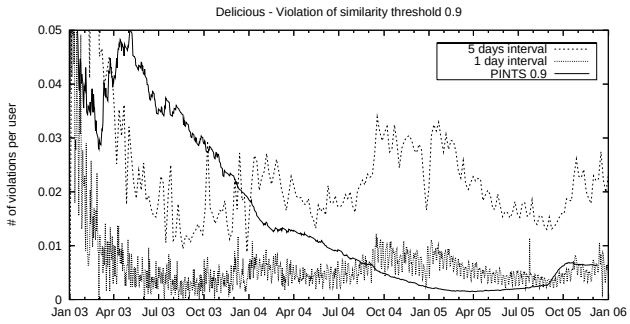
$\delta = 0.9$) is two orders of magnitude lower than for interval updates at the comparable level of accuracy (interval 1-day).

Summary. The presented results can be summarized as follows. The PINTS approach allows for flexible tuning of the tradeoff between the approximation accuracy for feature vectors of peers, and the resulting communication overhead. The predictive model for evolution of feature vectors helps to drastically reduce the number of required update messages at the very reasonable price of additional storage overhead on particular index peers for coefficients of the approximation function. At the same level of approximation accuracy, the PINTS approach causes a substantially lower message complexity than interval-based updates and instant propagation of changes. On the other hand, at the same level of message complexity, its accuracy is substantially higher. These results were systematically reproduced for both large-scale evaluation datasets (Flickr and del.icio.us) at different experimental settings.

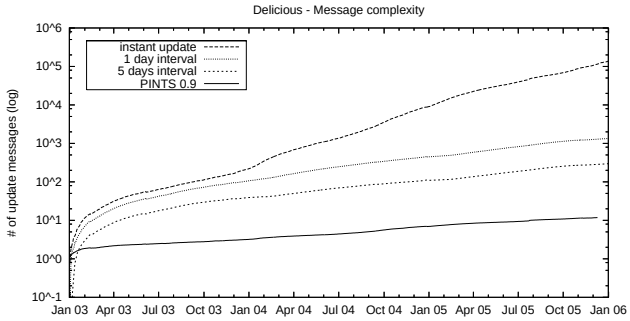
5. RELATED WORK

The high adoption rate of tagging systems has spurn of a large number of research efforts in order to understand tagging behavior and to improve access to data found in such tagging systems. For instance, [4] have proposed folkrank, a PageRank like mechanism for recommending resources. In this paper, we have focused on approaches based on the vector space model that are suitable for DHT-based P2P systems.

An important problem in peer-to-peer systems is the cardinality estimation for item sets, which is necessary for constructing feature weights in our approach. One solution is to directly exploit the underlying network structure [3, 7],



(a) Relative frequency of violations of the similarity threshold $\delta = 0.9$ with different update strategies



(b) Message complexity for different update strategies

Figure 3: Results for the Del.icio.us data set

as with distributed hashtables. However, tracking a huge number of cardinalities cannot be efficiently implemented in such a way. A different type of counting is the gossip-based approach [5, 6] where respective count information is exchanged iteratively between peers. In a stable network the count information converges toward the exact value. Albeit its good resilience to network changes it is not applicable for the highly dynamic tagging scenario with a huge number of required counters for particular tags.

Sampling-based counting algorithms [2] query a random subset of all peers to estimate the item frequency or derive histograms. A recent approach[9] combines sampling with gossipping to increase the accuracy. However, the results of these methods are not accurate for infrequently used tags.

A cardinality estimation method that fits well with distributed hashtables is the probabilistic counting with distributed hash sketches [10]. Our solution uses this approach for estimating the total number(s) of resources, users, or tags in the tagging environment with corresponding DHT-based index structures.

6. CONCLUSION AND FUTURE WORK

In this proposal we addressed the problem of constructing and maintaining reliable feature vectors for characterization of users and resources in a decentralized tagging environment. We adopted fundamental ideas of the IR-like vector space model and defined the notion of feature vectors for representing users, resources and tags in the collaborative tagging environment.

In a decentralized setting, cardinality estimations for computing particular features would require a noticeable amount of additional communication between peers in order to reconstruct missing global statistics. At this point, our solution PINTS aims to avoid unnecessary high network traffic by constructing predictive estimators that capture the evolution of the tagging framework. As a result, the required communication overhead for maintaining approximated feature vectors of users is substantially reduced.

In the future, we will conduct further refinements of the PINTS prediction model by using polynomial approximation instead of linear regression, capturing significant correlations between evolution of statistics for particular tags, and providing formal probabilistic quality guarantees for the results achieved so far. Our long-term objective is a P2P tagging system with reliable, efficient, and effective search and recommendation algorithms.

Acknowledgements. We thank Klaas Dellschaft and the Tagora Project (*http://tagora-project.eu*) for providing us with the Flickr and Del.icio.us reference data sets.

7. REFERENCES

[1] Peersim peer-to-peer simulator. (http://peersim.sf.net/).

[2] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating aggregates on a peer-to-peer network. Technical report, Computer Science Dept., Stanford University, 2003.

[3] Keren Horowitz and Dahlia Malkhi. Estimating network size from local information. *Inf. Process. Lett.*, 88(5):237–243, 2003.

[4] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In York Sure and John Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *LNAI*, pages 411–426, Heidelberg, June 2006. Springer.

[5] Márk Jelasity and Alberto Montresor. Epidemic-style proactive aggregationin large overlay networks. In *Proceedings of The 24th International Conference on Distributed ComputingSystems (ICDCS 2004)*, pages 102–109, Tokyo, Japan, 2004. IEEE Computer Society.

[6] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, page 482, Washington, DC, USA, 2003. IEEE Computer Society.

[7] Dionysios Kostoulas, Dimitrios Psaltoulis, Indranil Gupta, Ken Birman, and Al Demers. Decentralized schemes for size estimation in large and dynamic groups. In *NCA '05: Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications*, pages 41–48, Washington, DC, USA, 2005. IEEE Computer Society.

[8] R. Lambiotte and M. Ausloos. Collaborative tagging as a tripartite network. ArXiv Computer Science e-prints, December 2005.

[9] L. Massoulié, E. Le Merrer, A.M. Kermarrec, and A. Ganesh. Peer counting and sampling in overlay networks: random walk methods. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, New York, NY, USA, 2006. ACM.

[10] N. Ntarmos, P. Triantafillou, and G. Weikum. Counting at large: Efficient cardinality estimation in internet-scale data networks. In *Proceedings of the 22nd International Conference on Data Engineering*, page 40, Washington, DC, USA, April 2006. IEEE Computer Society.

[11] C. Schmitz, A. Hotho, R. Jäschke, and G. Stumme. Mining association rules in folksonomies. In *Proceedings of the 10th Conference on Data Science and Classification IFCS*, pages 261–270, Ljubljana, July 2006. Springer.

[12] S. Sizov and S. Siersdorfer. Towards social recommender systems for collaborative web 2.0 applications. In *Technical Report, University of Koblenz, available at http://www.uni-koblenz.de/FB4/Publications/Reports*, November 2007.