Building Directories for Social Tagging Systems

Denis Helic
Knowledge Management Institute
Graz University of Technology
Graz, Austria
dhelic@tugraz.at

Markus Strohmaier
Knowledge Management Institute and
Know-Center
Graz University of Technology
Graz, Austria
markus.strohmaier@tugraz.at

ABSTRACT

Today, a number of algorithms exist for constructing tag hierarchies from social tagging data. While these algorithms were designed with ontological goals in mind, we know very little about their properties from an *information retrieval* perspective, such as whether these tag hierarchies support efficient *navigation* in social tagging systems. The aim of this paper is to investigate the usefulness of such tag hierarchies (sometimes also called folksonomies - from folk-generated taxonomy) as directories that aid navigation in social tagging systems. To this end, we simulate navigation of directories as decentralized search on a network of tags using Kleinberg's model. In this model, a tag hierarchy can be applied as background knowledge for decentralized search. By constraining the visibility of nodes in the directories we aim to mimic typical constraints imposed by a practical user interface (UI), such as limiting the number of displayed subcategories or related categories. Our experiments on five different social tagging datasets show that existing tag hierarchy algorithms can support navigation in theory, but our results also demonstrate that they face tremendous challenges when user interface (UI) restrictions are taken into account. Based on this observation, we introduce a new algorithm that constructs efficiently navigable directories on our datasets. The results are relevant for engineers and scientists aiming to improve navigability of social tagging systems.

Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext/ Hypermedia—*Navigation*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Information networks*; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*Collaborative computing*

General Terms

Experimentation, Measurement, Algorithms

Keywords

Navigation, Simulation, Social Tagging, Folksonomy

CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.

1. INTRODUCTION

Social tagging systems allow users to organize information in an unconstrained manner by freely choosing so-called *tags* to annotate a set of potentially diverse resources, such as photos in Flickr, Web pages in Delicious or scientific articles in CiteULike [11]. In addition to using tagging systems for personal organization of information, users can *share* annotations with each other.

In past research, our research community has aimed to capture emergent semantic structures from the vocabulary originating in such systems, for example in the form of tag hierarchies [3, 4, 8, 12, 13] or so-called "folksonomies[1]" – from folk-generated taxonomies. However, this line of research has mainly pursued an ontological goal, and did not focus on the usefulness of folksonomies for *navigation*. For example [3, 12, 23] propose algorithms for constructing semantically sound tag hierarchies from social tagging data. A detailed analysis of approaches to semantic relatedness of tags in social tagging systems can be found in e.g. [7]. In our own previous work [19, 20], we investigated the extent to which tag semantics are influenced by user motivation and usage practices.

While there are a number of studies on the usefulness of tags for general information retrieval purposes, we know very little about the usefulness of tag hierarchies as *directories*, i.e. to support activities such as *browsing* or *navigation*. Recent theoretical work in this area for example has shown that the ability of tags to efficiently encode resources for later retrieval decreases over time [9]. In our own previous work [10], we evaluated the suitability of different tag hierarchies to support navigation in social tagging systems on a theoretical level - not taking user interface constraints into account. The practicability and usefulness of integrating tag hierarchies into user interfaces of social tagging systems thus remains unclear, and - to the best of our knowledge - has not been investigated so far.

This paper sets out to address this gap. We aim to assess the usefulness of tag hierarchies for supporting navigation processes in user interfaces of social tagging systems by adopting simulation as our method of choice. We start by hypothetically inserting excerpts of tag hierarchies into user interfaces of tagging systems similar to the way concept hierarchies are used in other systems, such as the Google Directory, to guide users to resources (see Figure 1 for an example). We then model a user navigating this hypothetical user interface: The user starts at an arbitrary tag and navigates to another tag in the system using the directory. Adopting Kleinberg's decentralized search algorithm, we simulate this by using tag hierarchies as background knowledge that guides a navigator towards the destination tag. In addition, we model typical user interface restrictions through a series of adjustable parameters such as limiting the number of subcategories presented to a user or constraining the number of related categories (please see Figure 1). By doing simulations,

[1]http://www.vanderwal.net/folksonomy.html

Figure 1: Google Directory user interface with typical directory elements including breadcrumbs, subcategories, and related categories.

we are able to conduct a thorough analysis of the effects of a variety of user interface restrictions on the usefulness of tag hierarchies for navigating 5 different real-world tagging datasets. This allows us to obtain novel insights and draw general conclusions about the drawbacks and promises of applying tag hierarchies as directories for navigating social tagging systems. We will show that tag hierarchies can support efficient navigation in social tagging systems in theory, but common user interface restrictions impair their practical suitability as a navigational aid significantly. Based on these results, we devise a novel directory algorithm capable of operating efficiently under typical user interface constraints.

In summary, this paper makes the following contributions:

- First, we introduce a framework for simulating navigation using different tag hierarchies in a social tagging system.
- Second, we show that although tag hierarchies can support efficient navigation in theory, under typical user interface restrictions (such as limiting the display of sub-categories) their ability to guide navigation is significantly impaired.
- Finally, we present a novel adaption of an existing algorithm [3] that restores efficient navigation under typical user interface restrictions.

The rest of this paper is organized as follows. In Section 2 we present a model for directory-based navigation using Kleinberg's algorithm for decentralized search in networks. In Subsection 2.3, we introduce the parameters for modeling user interface restrictions. In Section 3, we present our experimental setup and in Section 4 we discuss our experiment results. In Section 5 we introduce a new adaptation of an existing tag hierarchy algorithm and demonstrate how it can operate successfully under typical user interface constraints. Finally, we conclude the paper and provide some directions for future work.

2. MODELING DIRECTORY-BASED NAVIGATION

To investigate the implications of adopting tag hierarchies as a navigational aid in tagging systems, we model and simulate such navigation. Instead of observing real user behavior, we choose simulation mainly because current tagging systems do not offer directory support yet and simulation provides us with better experimental control and thus makes it possible to analyze different user interface issues across multiple datasets. In the following, we shortly introduce our simulation model and its theoretical background.

2.1 Greedy Search and Network Navigability

One of the research questions attracting a lot of interest in the field of networks is the relation between network structure and function, such as the relation between the structure and routing function of a network. Ever since the "small world" experiment [26] conducted by Stanley Milgram, researchers have been intrigued by the *routing efficiency* or *navigability* question in social networks – how people are able to find unknown people who are, potentially, geographically and socially distant to themselves. The key aspect of this question is the *absence of the global knowledge of the network* – people know only their friends and therefore *posses only the local knowledge of the network* but are still able to find unknown people. Similar navigability has been observed in other real networks such as metabolic or neural networks, or has been an important design goal for engineers of communication networks such as the Internet or different peer-to-peer networks (see e.g. [2]). Researchers identified the concept of *similarity between nodes* [22, 24, 32] or more generally the concept of *distance between nodes* [1, 15, 17, 18, 32] as an important aspect of establishing networking navigability. Combining the notion of distance between nodes with the algorithmic term of *greedy routing* [17], Kleinberg theoretically introduced network navigability [15, 18] in the following way: nodes use distance to select the next node in a routing session and the greedy algorithm selects the adjacent node closest (with the smallest distance) to the current destination node. The algorithm and its applications have been studied in the recent literature, see e.g. [16].

In [30] the authors abstract the notion of distance as introduced by Kleinberg to a *hidden distance* between nodes. Hidden distances define a *hidden metric spaces* which governs not only routing in the network but also the network formation and emergence of network structural properties such as power-law degree distributions and high node clustering. The authors connect observable emergent structural properties of a network with its navigability by defining a region of navigable networks in two dimensional space with clustering-coefficient [33] and power-law exponent as dimensions. On the other hand, a hidden metric space is also a geometric entity in which nodes are identified by their co-ordinates in it – distance between nodes is their geometric distance in that particular metric space. An interesting research question is the structure of such hidden metric spaces that underlie observable networks. In [5], the authors introduce a model with the circle as a hidden metric space and show its effects on routing in the global airport network. In [21] the authors discuss hyperbolic geometry as a hidden metric space whereas in [6] the authors apply hyperbolic geometry as a model of the hidden metric space of the Internet and design a novel greedy Internet routing algorithm.

The relation between Kleinberg's node distance and the recent work on hidden metric spaces can easily be established. In Kleinberg's model, the nodes are organized into a hierarchy according to their similarity – the distance between two nodes corresponds then to the height of their least common ancestor in that hierarchy [18] (Adamic [1] and Watts [32] introduce similar distance definitions). Hyperbolic geometry, as well as a hierarchy, distributes distances exponentially – it is, therefore, possible to approximate a hyperbolic metric space by a tree [21].

2.2 Simulating Navigation

We adopt a a tripartite hypernetwork model for social tagging systems where $V = R \cup U \cup T$, and R is the resource set, U is the user set, and T is the tag set [8, 27, 29]. An annotation of a particular resource with a particular tag produced by a particular user is a hyperedge (r, t, u), connecting three nodes from these three disjoint sets. Such a tripartite hypernetwork can be mapped onto three different bipartite networks connecting users and resources, users and tags, and tags and resources, or onto e.g. tag-to-tag networks. For different purposes it is often more practical to analyze one or more of these networks. For example, in the context of ontology learning, the bipartite networks of users and tags has been shown to be an effective projection [25]. In this paper, we focus on tag-to-tag networks (based on a tag-to-resource network) to mimic a tag-based user navigation task.

In a next step, we use a given tag hierarchy as a particular incarnation of a hidden metric space. We simulate greedy navigation through the observable tag-to-tag network querying the tag hierarchy for node distances – the idea is that the tag hierarchy is *useful* for supporting navigation *if* the greedy navigation using that hierarchy as the hidden metric space is successful.

The simulation algorithm starts its search at a randomly selected *start node* and tries to reach a randomly selected *destination node*. Search is carried out by moving along the links in the network in a number of intermediate steps. At each step, the link leading to the node that is closest (the node that is at the smallest hierarchy distance) to the destination node is selected. The basic idea behind such a greedy strategy is that there is a high probability to find a link to the destination node in the neighborhood of related nodes. The distance between two nodes v and w is calculated via a linear transformation of the height $h(v, w)$, which is the height of the least common ancestor of v and w in the tag hierarchy. More specifically, the parent node, the children nodes and the sibling nodes are considered to be at distance $d = 1$. From there on, the distance is recursively assigned, e.g., the parent's siblings and the grandchildren nodes are at distance $d = 2$, the children of the parent's siblings are at distance $d = 3$, the grandchildren of the parent's siblings are at distance $d = 4$ and so on. More details and an illustrative example of this particular simulation algorithm can be found in [10]. The algorithm is a variant of the simulation algorithm presented in [1].

For evaluation, we use the *length of the shortest path*. This reflects a typical scenario of navigation in tagging systems – the user will explore the tagging system by navigating to find relevant topics *as quickly as possible*, i.e., with as few clicks as possible. To model users behavior during navigation we apply the following strategy: if the agent arrives at a certain node for the second time, the search stops and is counted as a failure (no backtracking) – this mimics the situation where a user arrives at a tag that she has already visited, and then decides to, e.g., switch to the search field or to leave the system. In addition, if possible we try to avoid coming to a given node for the second time to model user exploration of alternative paths in the network. Thus, if the agent has two or more available paths with nodes at the same distances from the destination than a node that has not been already visited is preferred by the agent.

We simulate navigation in a large number of routing sessions (e.g. 1,000,000 sessions, see Section 3.3 for details on the experimental setup). Then, we quantify the quality of navigational support provided by a given tag hierarchy by measuring the success rate s of the greedy algorithm (the number of successfully reached destination nodes divided by the total number of routing sessions), and by the stretch τ [21], which is the ratio of the average greedy hops $\bar{h}$ to average shortest paths $\bar{l}$ (this measure tells us how longer are greedy paths as compared to global shortest paths). Those lengths are calculated over the e.g. 1,000,000 selected sessions. The measures are similar to those introduced in [6]. In addition to the global values calculated in [6], we calculate the measures for each observable shortest path in the networks.

In a final step, we analyze the structure of the greedy paths. This analysis provides insight into the importance of tag hierarchies and network nodes for the navigation process - in this way we can identify potential bottlenecks in folksonomies.

	BibSon.	CiteULike	Delicious	Flickr	LastFm
Tags	56,424	347,835	380,979	395,329	281,818
Links	2,003,986	27,536,381	39,808,439	17,524,927	84,787,780

Table 1: Dataset statistics.

2.3 UI Elements and Restrictions

Today, none of the popular tagging systems provide directory-based navigational aids. We therefore synthesize a model of a hypothetical user interface using user interface ideas from existing information systems with hierarchical organization of resources, such as Google Directory[2]. Such systems typically display the following user interface elements for any given node in a concept hierarchy (see Figure 1 for a screenshot or e.g. Google Directory[3]):

- **Breadcrumbs** provide the complete path to the root category.
- **Subcategories** provide links to more fine-grained categories. This list is typically limited in length. The user interface restricts how many subcategories are visible. We model this restriction by a parameter n.
- **Related categories** provide links to related categories. The user interface restricts how many related categories are visible. We model this second restriction by a parameter m.[4].

3. EXPERIMENTAL SETUP

We perform our experiments on five different tagging datasets. Directories are produced by a standard tag similarity network algorithm (see Section 3.2). General statistics about our datasets are depicted in Table 1, more details are presented next.

3.1 Datasets

Data from the following social tagging systems was used as an empirical basis:

- **Dataset BibSonomy**: This dataset[5] contains 916,495 annotations and 235,340 unique resources (scientific articles) from a dump of BibSonomy [14] until 2009-01-01. The tag-tag network comprises 56,424 nodes and 2,003,986 links.
- **Dataset CiteULike**: This dataset contains 6,328,021 annotations and 1,697,365 unique resources (scientific articles) and is available online[6]. The tag-tag network consists of 347,835 tags and 27,536,381 links.

[2]formerly DMOZ
[3]http://www.google.com/dirhp?hl=en
[4]In our model siblings of a particular node are considered to be related to that node
[5]http://www.kde.cs.uni-kassel.de/ws/dc09/
[6]http://www.citeulike.org/faq/data.adp

- **Dataset Delicious**: This dataset is an excerpt from the PINTS experimental dataset[7] containing a systematic crawl of Delicious and Flickr in 2006 and 2007. We extracted all data from November 2006. The resources in this dataset are Web addresses. The tag-tag network consists of 380,979 tags and 39,808,439 links.

- **Dataset Flickr**: This dataset is also an excerpt from the PINTS Flickr crawls. It contains the data from December 2005. The resources in Flickr are user-generated photos. The tag-tag network consists of 395,329 tags and 17,524,927 links.

- **Dataset LastFm** This dataset is from [28]. It contains annotations that were crawled from the last.fm website in the first half of 2009. The resources in this dataset are songs, artists and albums. The tag-tag network consists of 281,818 tags and 84,787,780 links.

3.2 Tag Hierarchy Algorithms

In previous work we have evaluated a series of existing tag hierarchy algorithms on a theoretical level, without taking user interface constraints into account [10]. As we have found that centrality-based algorithms outperform hierarchical clustering algorithms by a large margin (see [10, 31] for more details) we select one of these algorithms to conduct further investigations of their usefulness under typical user interface constraints. It is reasonable to assume that other centrality-based tag hierarchy algorithms will behave similarly under our constraints.

In this work we select an algorithm based on [12], where the authors propose to use so-called *tag similarity networks* and the centrality of tags for the construction of tag hierarchies. In their work, tag similarity networks are unweighted networks where each tag is a node in the network, and two nodes are linked to each other if their similarity is above a predefined similarity threshold. In the simplest case, the threshold is defined by tag overlap – if the tags do not overlap in at least one resource then they are not linked to each other in the tag similarity network. To construct tag hierarchies, the algorithm requires the ranking of nodes in a descending order according to how central the tags are in the tag similarity network. In particular, this ranking produces a centrality order where the most general tags from a dataset are in the top positions. The algorithm starts by a single node tree with the most general tag as the root node. The algorithm then proceeds by iterating through the centrality list and adding each tag to the tree – the algorithm calculates the similarities between the current tag and each tag currently present in the tree and adds the current tag as a child to its most similar tag. The authors define their algorithm in an extensible manner by supporting different similarity as well as different centrality measures. The presented algorithm uses cosine similarity and closeness centrality.

The work by [3] extends [12] by using tag co-occurrence as the similarity measure and the degree centrality as the centrality measure. In particular, the algorithm executes an extensive preprocessing of the dataset e.g. to remove synonym tags or to resolve ambiguous tags. For our work, we study this variation of the algorithm as both alternatives produce semantically and pragmatically comparable tag hierarchies. For reasons of simplicity, we skip the preprocessing of the dataset in this work and apply the alternative similarity and centrality measures only.

[7]https://www.uni-koblenz.de/FB4/Institutes/IFI/AGStaab/Research/DataSets/PINTSExperimentsDataSets/

3.3 Simulation Setup

With greedy navigation, we model and then simulate navigation in tagging systems. We select 1,000,000 distinct tag pairs that we call *search pairs* containing tags with degrees smaller or equal to 5 from the connected component of tag-to-tag tagging network. We adopt this low-degree selection strategy to make the simulated navigation task more "difficult" as finding e.g. the most central tag in a power law tag network is a trivial task. The first tag from the tag pair represents a *starting node* for navigation, modeling an arbitrary user entry page into the system (e.g. a landing page from a search engine, the latest tag from a news feed, or similar). We assume that users who come to the tagging system would *explore* the system to find one or more related topics or resources of current interest - we model this by means of the second tag (and the corresponding resource set) from the tag pair. We define this second tag as the *destination node* for the greedy navigation. The goal of the agent is to find a short path from the starting node to the destination node from the search pair.

To quantify the performance of the greedy navigator, we compare its path to the global shortest path (the shortest paths between two pairs assuming global knowledge of the network topology) between two selected nodes. We calculate the global shortest path between nodes from each search pair using breadth first search.

4. RESULTS

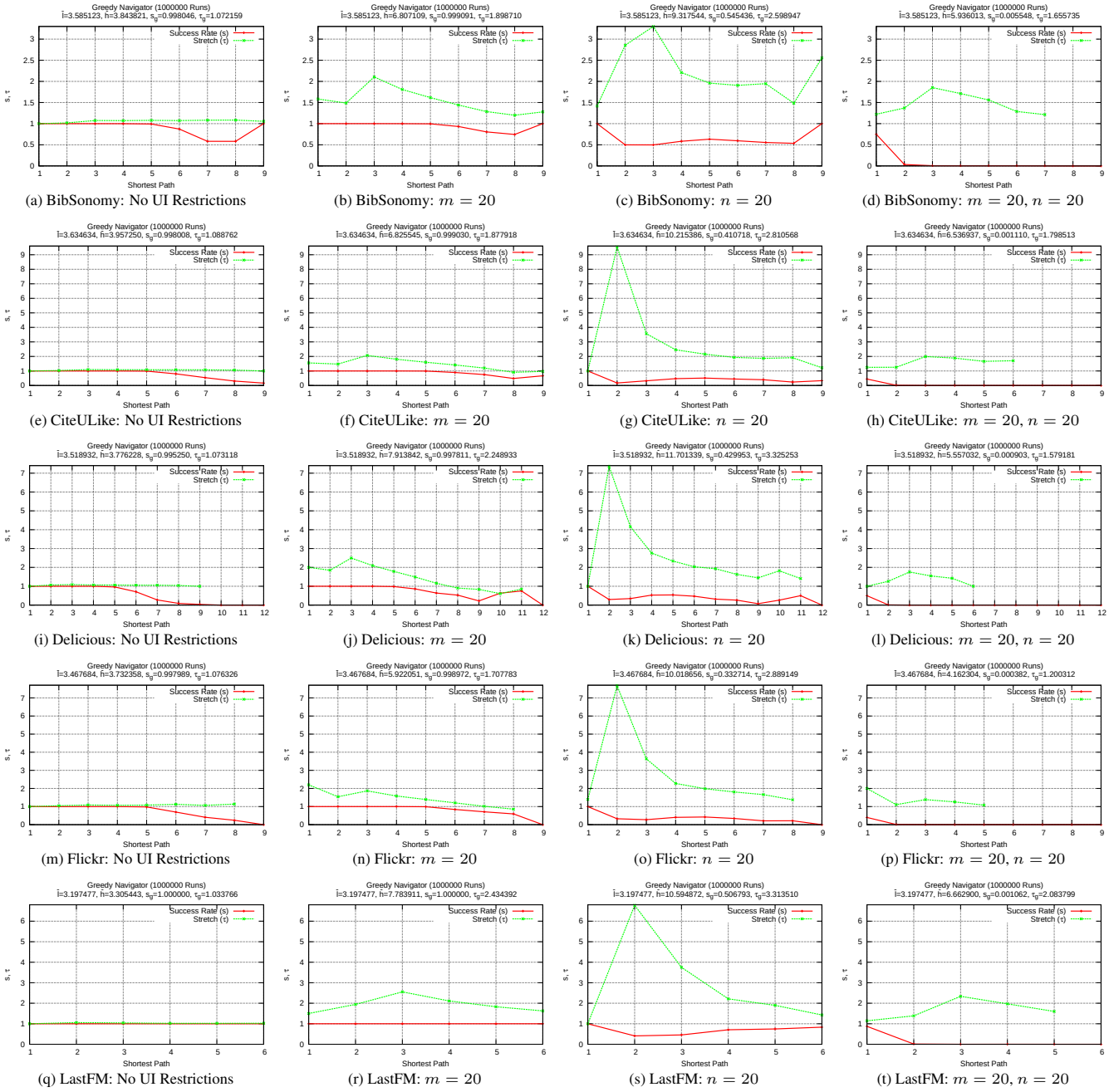
4.1 Navigational Efficiency of Tag Hierarchies

In this section we present the simulation results of adopting tag hierarchies as a navigational aid without any user interface restrictions, i.e. the entire tag hierarchy is accessible to a greedy navigator. It is important to note that this is a hypothetical setting, where a navigator has the entire directory at her disposal. In current directory-based systems (such as Google Directory), this is not the case. Therefore, these results are only of a theoretical value - they shed light on the nature of the greedy navigation process and help to identify potential pitfalls and bottlenecks, as well as ideas for solutions to those problems. Please note that these results are consistent with our previous results on theoretical suitability of tag hierarchies as a navigational support [10] – where here, we extend this investigation by discussing the structure of paths of the greedy navigator.

Consistently over all datasets, the greedy navigator performs very well with success rates close to 100%. Moreover, the navigator is very efficient in finding a short paths - in most cases it requires the same number of steps as the global shortest path - the global stretch is very close to 1 (see the far-left column of Figure 2).

As observed by [5], the path structure of the greedy navigator follows the zoom-out/zoom-in phase pattern. In the zoom-out phase, the navigator starts at a low degree node in the network periphery; it then exits the periphery and enters the network core visiting one of the high degree nodes there. In the zoom-in phase, the navigator leaves the core for its low-degree destination node in the periphery. Over all datasets, the top nodes are the most visited nodes - these are the nodes from the network core where the phase transition in the navigation process occurs (see the far-left column of Figure 3).

Table 2 shows the importance of the top degree nodes for greedy navigation – a huge fraction of all greedy paths go through the top 10% tags. This can be characterized as the main pitfall of tag hierarchies as a navigational tool. The top level categories in tag hierarchies have a huge number (hundreds, even thousands) of subcategories and related categories (see e.g. [10] or Figure 4a). The greedy navigator has no problems in finding the optimal path towards the network periphery from such a high-degree node – all



(a) BibSonomy: No UI Restrictions (b) BibSonomy: $m = 20$ (c) BibSonomy: $n = 20$ (d) BibSonomy: $m = 20$, $n = 20$

(e) CiteULike: No UI Restrictions (f) CiteULike: $m = 20$ (g) CiteULike: $n = 20$ (h) CiteULike: $m = 20$, $n = 20$

(i) Delicious: No UI Restrictions (j) Delicious: $m = 20$ (k) Delicious: $n = 20$ (l) Delicious: $m = 20$, $n = 20$

(m) Flickr: No UI Restrictions (n) Flickr: $m = 20$ (o) Flickr: $n = 20$ (p) Flickr: $m = 20$, $n = 20$

(q) LastFM: No UI Restrictions (r) LastFM: $m = 20$ (s) LastFM: $n = 20$ (t) LastFM: $m = 20$, $n = 20$

Figure 2: Greedy Navigator Success Rate (s) and Stretch (τ) - overall values and distributions per observable shortest path. *Far-Left:* (No UI restrictions) Theoretical evaluation of folksonomies as a navigational aid gives excellent results – both success rate and stretch are almost ideal. Slightly worst success rates for longer shortest paths (e.g. $l \geq 6$) are caused by poor connectivity and low degrees of the destination nodes and nodes leading to them. *Center-Left:* (Limited display of related categories with $m = 20$) Instead of taking a shortcut between related nodes the navigator needs to take a longer path over the parent node. Since less shortcuts between related nodes are being taken the number of navigator hops increases resulting in increased stretch values. On the other hand, the success rate remains almost untouched. *Center-Right:* (limited display of subcategories with $n = 20$) Under this restriction the navigator in many cases can not find the proper way towards the destination node in the periphery. The appropriate preceding nodes are very often not visible from a given high-degree node and this results in a failure. When successful the navigator bounces between the related and parent nodes, or between siblings until it finds the right path, see e.g. high stretch values for the shortest path of 2 (in a typical case we have here two low-degree siblings related via a high degree parent). *Far-Right:* ($m = 20$ and $n = 20$). Considering practical user interface restriction folksonomies are useless for supporting navigation. The success rate drops below 1%.

TopN(%)	Visits(%)
Root	4.94
1%	14.69
5%	33.99
10%	44.11

(a) BibSonomy

TopN(%)	Visits(%)
Root	4.18
1%	17.73
5%	30.01
10%	36.67

(b) CiteULike

TopN(%)	Visits(%)
Root	0.48
1%	19.83
5%	37.92
10%	45.38

(c) Delicious

TopN(%)	Visits(%)
Root	6.37
1%	20.09
5%	38.30
10%	46.27

(d) Flickr

TopN(%)	Visits(%)
Root	4.66
1%	35.80
5%	56.63
10%	62.74

(e) LastFM

Table 2: Percentage of greedy visits to the network core, i.e. visits to the (percentage of) the top degree nodes. Large proportion (36%-62%) of all greedy paths visit some of the top 10% nodes. This characterizes the two phase greedy navigation process - in the first "zoom-out" phase the greedy navigator exits the periphery and enters the network core whereas in the second "zoom-in" phase the greedy navigator leaves the core for its final destination in the network periphery.

paths at any given node are always *visible* to the greedy navigator. On the other hand, a typical user interface restricts visibility of subcategories to only a few, e.g. 10 or 20 categories which are sorted by a ranking algorithm. Typically, all other subcategories are only visible after clicking through e.g. a paginated category list.

In the next section we model such UI restrictions and investigate the *practical* usefulness of tag hierarchies to support navigation processes in user interfaces of social tagging systems.

4.2 Effects of UI Elements and Restrictions

In our simulations, we model the limited ability of user interfaces to display a directory in its entirety. This enables us to study the limitations of tag hierarchies where the number of subcategories is very high (in some cases, existing tag hierarchy algorithms produce nodes with thousands of children nodes). To model the constraints of a typical user interface, we set $n = 20$ and $m = 20$ and select randomly n tags as subcategories and respectively m tags as related categories. While similar constraints can be found in typical directory-based information systems such as Google Directory, our framework can easily evaluate and accommodate other values. For all practical purposes though, the results presented in this paper are based on the above parameter values.

Breadcrumbs. In all our experiments that model UI constraints, we introduce breadcrumb-aided navigation to our hypothetical user interfaces. In our simulations, breadcrumbs can speed up the zoom-out phase - in most cases the navigator jumps over low-degree nodes directly to the top nodes (see the center-left, center-right, and far-right columns in Figure 3). This is an intuitive behavior that can be expected from a greedy navigator.

In many cases, the effects of breadcrumbs on the greedy navigator is a decrease in the number of hops - often the navigator takes a direct, otherwise non-present, breadcrumb path to e.g. the root node. In some rare cases breadcrumbs can cause the stretch to even sink below 1 (see e.g. stretch values for $l \geq 8$ in 2j).

Subcategories. Constraining the display of subcategories to $n = 20$ impairs the success rate of the navigator tremendously. Destination nodes have small degrees - finding a path to those nodes requires the navigator to select appropriate preceding nodes from a given high-degree node. However, because of the particular user interface restriction, the appropriate nodes are very often not visible from a given high-degree node and this often leads to a failure. High stretch values indicate that the navigator bounces between the related and parent nodes until it finds its way down towards the destination node (see the center-right column of Figure 2).

Since the number of visible children is limited to a small value the zoom-in phase is disrupted - only few paths headed towards their final destination in the periphery find their way promptly. Instead, the navigator "bounces" between parent and sibling nodes to find its way down to the destination node (see the center-right column in Figure 3).

Related categories. The effect of limiting the number of related nodes (siblings) visible from the current node to $m = 20$ is that far less shortcuts between related nodes are being taken. Instead of taking a shortcut, the navigator often visits one of the parent nodes and only then one of the siblings or grand-siblings - this increases the number of necessary hops. Although breadcrumbs reduce the total number of hops the first effect is, by far, more significant - the total hops number and consequently the stretch increases (see the center-left column of Figure 2). On the other hand, the success rate remains almost uneffected. In certain cases it even increases (see e.g. 2b) - the navigator does not take the path over a non-visible sibling but takes an alternative (more successful) path via breadcrumbs or a visible sibling at the same distance to the destination node.

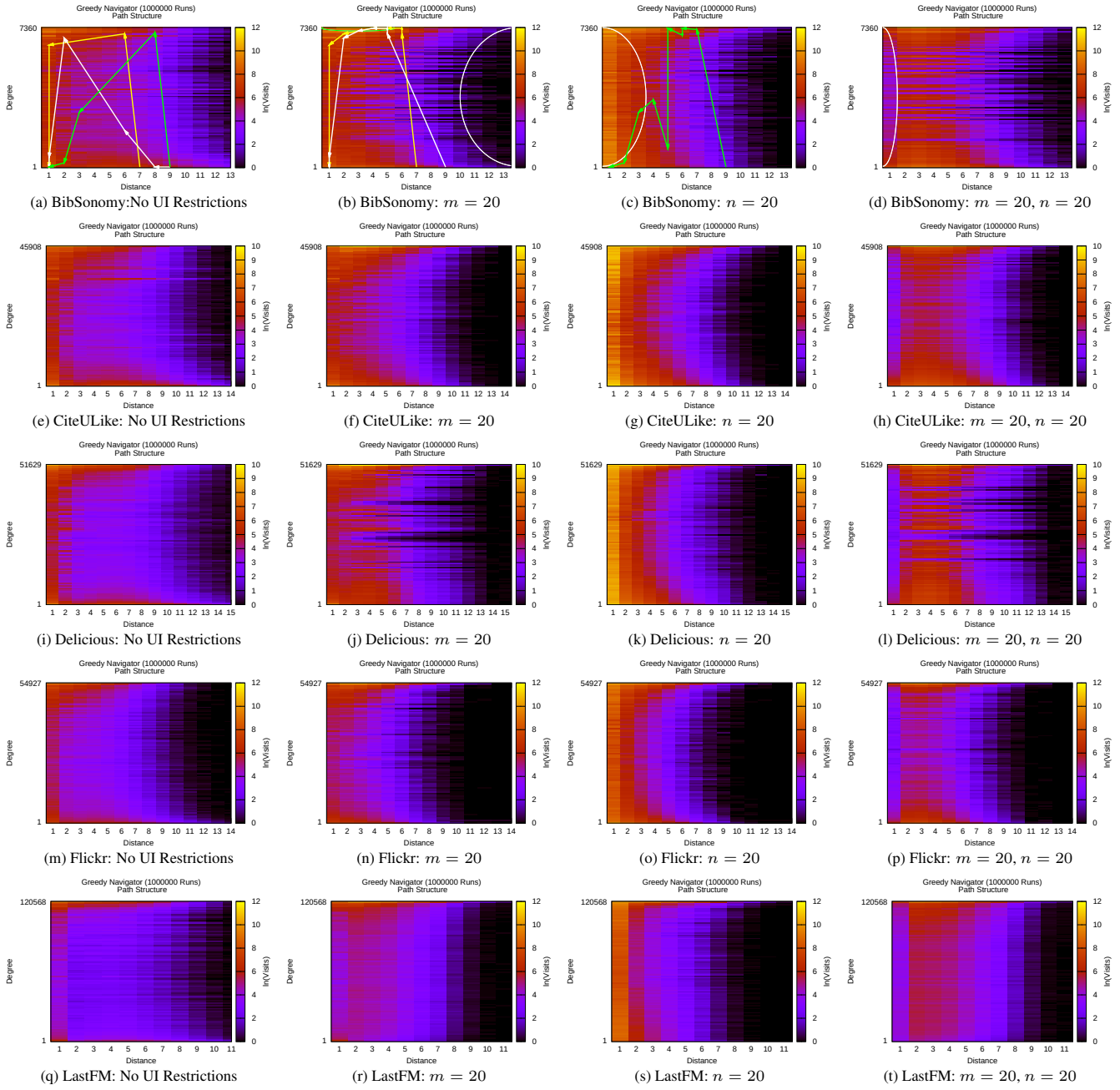
Concerning the path structure the following effects can be observed. While in the core, the majory of paths go up to the parent or root node. As the number of visible siblings is limited shortcuts are taken rarely (see the center-left column in Figure 3).

Related and subcategories restriction. When both user interface restrictions are applied at the same time (which again is typical in hierarchically organized information systems), the success rate drops drastically (see the far-right column in Figure 2). Even some of the nodes at $l = 1$ can not be found by the navigator - because of the user interface restrictions they are not visible anymore from the start node. On the one hand, the navigator can not find its way down towards the destination node. On the other, the "bouncing" between the related nodes is now also "forbidden" and this leads to disappointing success rates. Now - since both of the problems apply - the zoom-in phase is disrupted and the navigator does not "bounce" between siblings anymore. Most of the paths lead to the root node and then finish there without finding their way out of the core (see the far-right column in Figure 3).

In summary, these results suggest that existing algorithms for constructing tag hierarchies produce hierarchical structures that are not useful for aiding navigation under practical user interface constraints.

5. ALGORITHM

The main pitfall of the tag hierarchies that we simulated so far is their breadth – top level categories have too many subcategories, typically hundreds or thousands. Figure 4a depicts out-degree distributions of the tag hierarchies in our datasets. These distributions are essentially power-low distributions with a small number of high-degree tags (these are the top hierarchy nodes) and a large number of low-degree tags (these are the nodes from the lower hierarchy levels). As our previous analysis have shown the top level



(a) BibSonomy:No UI Restrictions (b) BibSonomy: $m = 20$ (c) BibSonomy: $n = 20$ (d) BibSonomy: $m = 20, n = 20$

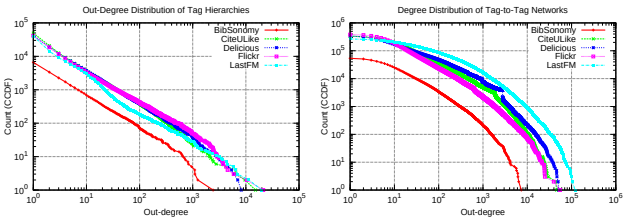
(e) CiteULike: No UI Restrictions (f) CiteULike: $m = 20$ (g) CiteULike: $n = 20$ (h) CiteULike: $m = 20, n = 20$

(i) Delicious: No UI Restrictions (j) Delicious: $m = 20$ (k) Delicious: $n = 20$ (l) Delicious: $m = 20, n = 20$

(m) Flickr: No UI Restrictions (n) Flickr: $m = 20$ (o) Flickr: $n = 20$ (p) Flickr: $m = 20, n = 20$

(q) LastFM: No UI Restrictions (r) LastFM: $m = 20$ (s) LastFM: $n = 20$ (t) LastFM: $m = 20, n = 20$

Figure 3: Greedy Navigator Path Structure. The density maps visualize visit frequency to nodes of a given degree at a given distance - the color is logarithm of the visit frequency (black indicating less visits; yellow indicating more visits). The zoom-out/zoom-in phase pattern can be observed across all experiments. *Far-Left:* (No UI restrictions) In 3a three typical paths are depicted with arrows. All paths start at low-degree nodes and long distances (the bottom-right side) proceeding to a core node in the zoom-out phase (the top-center position) and finishing again at a destination low-degree node (the bottom-left side). Many paths in the core proceed to the root node but many of those paths take also shortcuts between related high-degree siblings (see e.g. the yellow path in 3a). *Center-Left:* ($m = 20$) Because of breadcrumbs, the navigator skips low-degree nodes (cf. the white lines in 3a and 3b) and leaves the periphery immediately (more black/less violet in the white marked region in 3b). Same phenomenon can be observed in center-right and far-right diagrams. The majority of paths in the core go up to the root node. As the number of visible siblings is limited, shortcuts are taken rarely (greater numbers of yellow/orange pixels in the green marked region in 3b) increasing the total number of hops (cf. the yellow lines in 3a and 3b). *Center-Right:* ($n = 20$) The navigator "bounces" between parent and sibling nodes on its way down to the destination node (cf. the green lines in 3a and 3c). The number of mid-degree nodes visited at mid and small distances increases (more orange/yellow in the white marked region in 3c). *Far-Right:* ($m = 20$ and $n = 20$) The navigator is stuck at the root node without finding its way out of the core (more violet/less orange in the white marked region in 3d).



(a) Tag hierarchies (b) Tag networks

Figure 4: *Left:* Cumulative out-degree distribution of tag hierarchies produces by the algorithm described in [3]. Top level tags have hundreds, even thousands of subcategories – this makes those hierarchies useless for navigation as a typical user interface displays only a limited number of subcategories. *Right:* Cumulative degree distributions of tag-to-tag networks show existence of a large number of low-degree nodes that are typically connected to a single network hub.

nodes are crucial for the navigation. However, the UI limitations restrict the number of displayed subcategories to only a few and thus limit the usefulness of tag hierarchies as a navigational aid.

This requires a new tag hierarchy algorithm that is able to take into account display limitations of a typical user interface. In other words, the algorithm takes as an input parameter the display size and produces a tag hierarchy that can fit into a given user interface. Thus, the input parameter defines maximal number of subcategories and related categories that can be displayed. Technically, this input parameter defines branching factor b of the tag hierarchy.

In the following, we present a novel adaptation of the previously applied tag hierarchy algorithm that significantly recovers navigational support under typical user interface constraints by introducing a branching factor b as an input parameter. Moreover, we will demonstrate that *our algorithm produces tag hierarchies that efficiently support users in navigating social tagging systems*.

The new algorithm is a two-pass procedure. In the first pass the algorithm works similarly as proposed in [3] with the difference that the algorithm populates not only a single tree but a forest with multiple trees. The algorithm attaches maximally $b-1$ subcategories to any given category in the forest. Before the second pass the algorithm sorts the produced trees in descending order of their size (the number of categories that they contain). We will call the largest tree in the forest the main tree.

In the second pass the algorithm attaches other trees to the main tree by linking the root node of a given tree to its most similar node from the main tree. If the most similar node has only one free subcategory spot left then a special misc category is first inserted in that free spot and then the given tree is attached to that misc category. The nesting of misc categories is necessary and it can be arbitrarily deep, e.g. if a misc category has only a single free subcategory spot left then another sub-misc category is first inserted in that spot and then a given tree is attached to that sub-misc category.

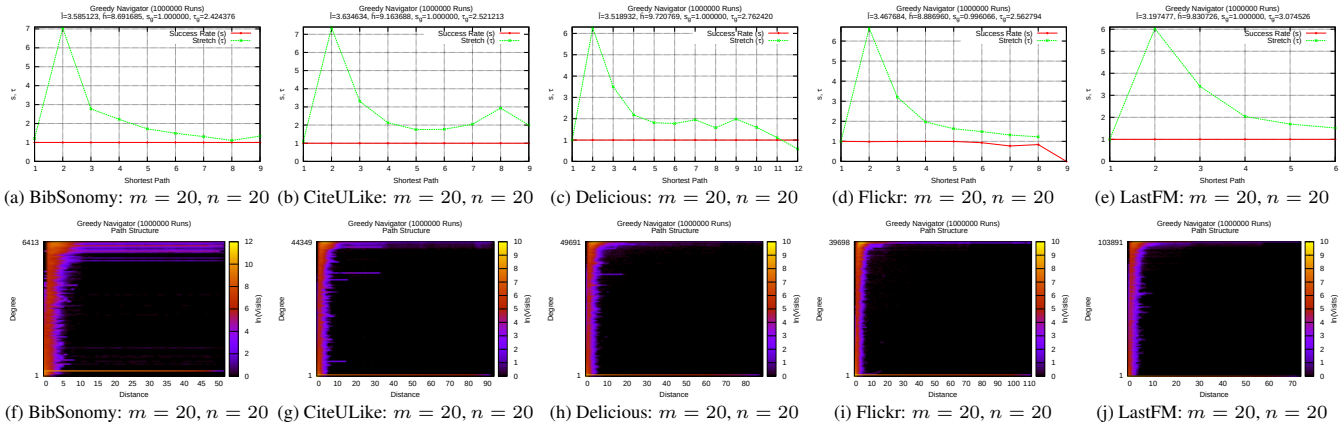
The goal of this approach is to connect as much trees as possible (and thus as much tags as possible) to the main tree without creating deep misc/sub-misc subtrees. However, such nested misc category constructs can not be avoided completely as the structure of the tag-to-tag network dictates their existence. Figure 4b shows the out-degree distributions of tag-to-tag networks in our analyzed datasets. The distributions show that there is a huge number of tags with a very small degrees – this means that these tags also have low centrality values. Further, these tags are typically connected only to high-degree tags. Because of their small centrality low-degree

Input: b **branching factor**
Output: $F =< V, E >$
add $C[0]$ to V
/ Start of the first pass */*
for $i = 1...|C|$ **do**
  $t \leftarrow C[i]$
  add t to V
  $attached \leftarrow false$
  $j \leftarrow 0$
  repeat
    $candidate \leftarrow S_t[j]$
    if $(getOutDegree(V, candidate) < (b-1))$ **then**
      add $(candidate, t)$ to E
      $attached \leftarrow true$
    end if
    $j \leftarrow j+1$
  until $(j < |S_t|)$ && $(attached == false)$
end for
/ End of the first pass */*
$WCCS \leftarrow getWccs(V)$
/ Start of the second pass, WCSS[0] is the main tree */*
for $i = 1...|WCCS|$ **do**
  $t \leftarrow getRoot(WCCS[i])$
  $attached \leftarrow false$
  $j \leftarrow 0$
  repeat
    $candidate \leftarrow S_t[j]$
    if $(candidate \in WCCS[0])$ **then**
      if $(getOutDegree(V, candidate) < b)$ **then**
        add $(candidate, t)$ to E
        $attached \leftarrow true$
      else
        addToMiscCategory(candidate, t)
      end if
    end if
    $j \leftarrow j+1$
  until $(j < |S_t|)$ && $(attached == false)$
  if $(attached == false)$ **then**
    addToMiscCategory(getRoot(WCCS[0], t)
  end if
end for
/ End of the second pass */*

Figure 5: Tag Hierarchy Algorithm

tags are therefore added to the final tag hierarchy in the later stages of the process – typically when all of the free spots of high-degree tags are already taken.

For example, we want to discuss the following situation. Suppose there is a high-degree node in the network. Because of its degree the node possesses high centrality and resides therefore in one of the top hierarchy levels. In a typical power-law network, such node is connected to a small number of other high or mid-degree nodes with higher centrality values and to a significantly larger number of low-degree nodes with small centrality values (the latter representing the long tail). Thus, the algorithm first attaches adjacent high and mid-degree nodes as subcategories of the given node and uses all free subcategory spots. With high probability, the low-degree neighbors have links only to the given node (or similar high-degree nodes with no free subcategory spots) and need to be attached to that node via its misc subcategory. As the number of



(a) BibSonomy: $m = 20, n = 20$ (b) CiteULike: $m = 20, n = 20$ (c) Delicious: $m = 20, n = 20$ (d) Flickr: $m = 20, n = 20$ (e) LastFM: $m = 20, n = 20$

(f) BibSonomy: $m = 20, n = 20$ (g) CiteULike: $m = 20, n = 20$ (h) Delicious: $m = 20, n = 20$ (i) Flickr: $m = 20, n = 20$ (j) LastFM: $m = 20, n = 20$

Figure 6: Top row: success rate and stretch of the improved algorithm. The new hierarchy is able to completely recover the success rate, however with increased stretch values. Bottom row: The density maps reveal a very clean zoom-out/zoom-in phase with zoom-out phase typically shortened by jumping over low degree nodes to the top level nodes via breadcrumbs. The long tail navigation is visible across all datasets as the orange/yellow line near the bottom of the diagrams.

such neighbors is larger than a typical branching factor nesting of misc categories is necessity.

Please note that the misc subtrees do not break semantics of the tag-to-tag networks, as no tags are attached to unrelated tags. Rather, the tags are pushed farther away from their most related tags into its nested misc categories.

The proposed algorithm (see Figure 5) requires:

- C is a array of tags sorted in descending order of their centrality in the tag-to-tag network.
- S is a array of arrays S_t. Each array S_t is sorted in descending order of similarity of tag t to other tags in the network.
- Function $getOutDegree(V, t)$ returns out-degree of node t in graph V.
- Function $getWccs(V)$ returns an array of weakly connected components of V sorted in descending order of components sizes. Function $getRoot(V)$ returns the root node of graph V.
- Function $addToMiscCategory(candidate, t)$ first creates a misc category for the $candidate$ tag and adds the tag t to that misc category. If the misc category is already full it will create a sub-misc category one level below in the hierarchy. The process is repeated recursively until a misc category with a free spot is found. In addition, the function *engineers* the tag-to-tag network linking each misc category to its parent, siblings, and children categories.

Our experiments with the hierarchies produced by the new algorithm show significant improvements in success rate across all datasets. Essentially, we were able to completely recover the success rate (see Figure 6). However, stretch increases and ranges between 2 and 3. This is caused by navigating the long tail of low degree tags – as those tags are situated within nested misc categories the number of hops before reaching the destination node increases with the number of hierarchy levels containing long tail tags. This phenomena is mostly visible for global shortest paths of 2 as two long tail tags are very often both linked to one and the same high-degree node.

6. CONCLUSION

While algorithms for constructing tag hierarchies have received much attention from a semantic research perspective, their usefulness for supporting *navigation* in social tagging systems under typical user interface constraints was generally unknown. The simulations and experimental results conducted in this work demonstrate that using existing tag hierarchies produced by existing algorithms as directories in user interfaces of social tagging systems yields sub-optimal results. Based on these insights, we presented and evaluated a novel adaptation of an existing algorithm that (i) significantly outperforms existing approaches, and (i) almost matches the results of the same algorithms in a hypothetical (i.e. unconstrained user interface) setting. The results of this work illuminate a way towards integrating tag hierarchies into user interfaces in an efficient way, i.e. in a way that enables users to navigate directories efficiently. Our work is relevant for engineers interested in integrating directory-based navigation support into user interfaces of social tagging systems and for scientists interested in the design and analysis of social tagging systems.

7. REFERENCES

[1] L. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187 – 203, 2005.

[2] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in power-law networks. *Physical Review E*, 64(4):046135 1–8, Sep 2001.

[3] D. Benz, A. Hotho, and G. Stumme. Semantics made by you and me: Self-emerging ontologies can capture the diversity of shared knowledge. In *Proc. of the 2nd Web Science Conference (WebSci10)*, Raleigh, NC, USA, 2010. Web Science Trust.

[4] D. Benz, C. Körner, A. Hotho, G. Stumme, and M. Strohmaier. One tag to bind them all : Measuring term abstractness in social metadata. In G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, J. Pan, and P. D. Leenheer, editors, *Proceedings of the 8th Extended Semantic Web Conference (ESWC 2011)*, Heraklion, Crete, May 2011.

[5] M. Boguñá, D. Krioukov, and K. C. Claffy. Navigability of complex networks. *Nature Physics*, 5:74–80, Jan. 2009.

[6] M. Boguñá, F. Papadopoulos, and D. Krioukov. Sustaining the Internet with hyperbolic mapping. *Nature Communications*, 1:62, Sept. 2010.

[7] C. Cattuto, D. Benz, A. Hotho, and G. Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, and K. Thirunarayan, editors, *The Semantic Web – ISWC 2008, Proc. of International Semantic Web Conference 2008*, volume 5318 of *LNAI*, pages 615–631, Heidelberg, 2008. Springer.

[8] C. Cattuto, C. Schmitz, A. Baldassarri, V. D. P. Servedio, V. Loreto, A. Hotho, M. Grahl, and G. Stumme. Network properties of folksonomies. *AI Commun.*, 20(4):245–262, 2007.

[9] E. H. Chi and T. Mytkowicz. Understanding the efficiency of social tagging systems using information theory. In *Proc. of the nineteenth ACM conference on Hypertext and hypermedia, HT '08*, pages 81–88, New York, NY, USA, 2008. ACM.

[10] D. Helic, M. Strohmaier, C. Trattner, M. Muhr, and K. Lerman. Pragmatic evaluation of folksonomies. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 417–426, New York, NY, USA, 2011. ACM.

[11] D. Helic, C. Trattner, M. Strohmaier, and K. Andrews. On the navigability of social tagging systems. In *Proc. of 2010 IEEE International Conference on Social Computing*, pages 161–168, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[12] P. Heymann and H. Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical Report 2006-10, Stanford InfoLab, April 2006.

[13] A. Hotho, R. Jaeschke, C. Schmitz, and G. Stumme. Folkrank: A ranking algorithm for folksonomies. In *Proc. FGIR 2006*, pages 111–114, Bonn, Germany, 2006. Gesellschaft Für Informatik.

[14] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Bibsonomy: A social bookmark and publication sharing system. In A. de Moor, S. Polovina, and H. Delugach, editors, *Proceedings of the Conceptual Structures Tool Interoperability Workshop at the 14th International Conference on Conceptual Structures*, pages 87–102, Aalborg, Denmark, July 2006. Aalborg University Press.

[15] J. Kleinberg. The small-world phenomenon: an algorithm perspective. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, STOC '00, pages 163–170, New York, NY, USA, 2000. ACM.

[16] J. Kleinberg. Complex networks and decentralized search algorithms. In *International Congress of Mathematicians (ICM)*, pages 1019–1044, Zürich, Switzerland, 2006. European Mathematical Society Publishing House.

[17] J. M. Kleinberg. Navigation in a small world. *Nature*, 406(6798):845, August 2000.

[18] J. M. Kleinberg. Small-world phenomena and the dynamics of information. In *Advances in Neural Information Processing Systems (NIPS) 14*, page 2001, Cambridge, MA, USA, 2001. MIT Press.

[19] C. Koerner, D. Benz, M. Strohmaier, A. Hotho, and G. Stumme. Stop thinking, start tagging - tag semantics emerge from collaborative verbosity. In *Proc. of the 19th International World Wide Web Conference (WWW 2010)*, Raleigh, NC, USA, Apr. 2010. ACM.

[20] C. Koerner, R. Kern, H. P. Grahsl, and M. Strohmaier. Of categorizers and describers: An evaluation of quantitative measures for tagging motivation. In *21st ACM SIGWEB Conference on Hypertext and Hypermedia (HT 2010), Toronto, Canada, ACM*, Toronto, Canada, June 2010.

[21] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguñá. Hyperbolic geometry of complex networks. *Phys. Rev. E*, 82(3):036106, Sep 2010.

[22] E. A. Leicht, P. Holme, and M. E. J. Newman. Vertex similarity in networks. *Phys. Rev. E*, 73(2):026120, Feb 2006.

[23] R. Li, S. Bao, Y. Yu, B. Fei, and Z. Su. Towards effective browsing of large scale social annotations. In *Proc. of the 16th international conference on World Wide Web, WWW '07*, page 952, New York, NY, USA, 2007. ACM.

[24] F. Menczer. Growing and navigating the small world web by local content. *Proc. Natl. Acad. Sci. USA*, 99(22):14014–14019, 2002.

[25] P. Mika. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):5–15, 2007.

[26] S. Milgram. The small world problem. *Psychology Today*, 1:60–67, 1967.

[27] M. Ramezani, J. Sandvig, T. Schimoler, J. Gemmell, B. Mobasher, and R. Burke. Evaluating the impact of attacks in collaborative tagging environments. In *Computational Science and Engineering, 2009. CSE '09. International Conference on*, volume 4, pages 136 –143, Los Alamitos, CA, USA, aug. 2009. IEEE Computer Society.

[28] R. Schifanella, A. Barrat, C. Cattuto, B. Markines, and F. Menczer. Folks in folksonomies: social link prediction from shared metadata. In *Proc. of the third ACM international conference on Web search and data mining, WSDM '10*, pages 271–280, New York, NY, USA, 2010. ACM.

[29] C. Schmitz, A. Hotho, R. Jöschke, and G. Stumme. Mining association rules in folksonomies. In *Data Science and Classification: Proc. of the 10th IFCS Conf., Studies in Classification, Data Analysis, and Knowledge Organization*, pages 261–270, Berlin, Heidelberg, 2006. Springer.

[30] M. A. Serrano, D. Krioukov, and M. Boguñá. Self-similarity of complex networks and hidden metric spaces. *Phys. Rev. Lett.*, 100(7):078701, Feb 2008.

[31] M. Strohmaier, D. Helic, D. Benz, C. Körner, and R. Kern. Evaluation of folksonomy induction algorithms. under review.

[32] D. J. Watts, P. S. Dodds, and M. E. J. Newman. Identity and search in social networks. *Science*, 296:1302–1305, 2002.

[33] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.