Christoph Schmitz

Self-Organized Collaborative Knowledge Management

kassel university press

This work has been accepted by the faculty of electrical engineering / computer science of the University of Kassel as a thesis for acquiring the academic degree of Doktor der Naturwissenschaften (Dr. rer. nat.).

Supervisor: Prof. Dr. Gerd Stumme
Co-Supervisor: Prof. Dr. Rudi Studer

Defense day: 12th July 2007

Bibliographic information published by Deutsche Nationalbibliothek
Die Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie;
detailed bibliographic data is available in the Internet at http://dnb.ddb.de

Zugl.: Kassel, Univ., Diss. 2007
ISBN: 978-3-89958-325-0
URN: urn:nbn:de:0002-325-9

Printed by: Unidruckerei, University of Kassel
Printed in Germany

At around four o'clock most Saturday afternoons, just when I make us all a cup of tea, I have a little glow on, maybe because this is after all my work, and it's going OK, maybe because I'm proud of us, of the way that, though our talents are small and peculiar, we use them to their best advantage.

Nick Hornby, "High Fidelity" (1995)

Contents

List of Figures

List of Tables

Danksagung

Zuerst möchte ich Prof. Gerd Stumme und Prof. Rudi Studer für die Betreuung und Begutachtung dieser Arbeit danken und für die spannende und lehrreiche Zeit, die ich in ihren Arbeitsgruppen verbringen konnte. Weiterhin danke ich Prof. Klaus David und Prof. Kurt Geihs für die Bereitschaft, als Prüfer in der Promotionskommission mitzuwirken.

Bei meinen Kollegen Andreas Hotho, Robert Jäschke, Miranda Grahl, Beate Krause sowie bei den Kollegen vom AIFB Karlsruhe bedanke ich mich für die gute Zusammenarbeit und viele angeregte Diskussionen nicht nur über Informatik. Silke Finis, Sven Stefani, Meike Siebert-Adzic, Monika Vopicka und Jörn Dreyer möchte ich für ihre Hilfe in allen weltlichen Dingen des Unialltags danken.

Meiner Familie gilt besonderer Dank für ihre Unterstützung in allen Lebenslagen, ohne die diese Arbeit niemals zustande gekommen wäre.

Abstract

In today's organisations, *knowledge* is regarded as the key economic resource. The acquisition, retaining, development, and use of knowledge resources is thus an important issue. Activities such as the ones named before which take influence on the organizational knowledge base are subsumed under the term *knowledge management*.

Knowledge management has often been regarded as a task which, similarly to, e. g., software engineering, can be planned and implemented in structured development processes. There is evidence, however, that these knowledge processes and meta processes often disregard the particular knowledge management requirements of each individual, and that organisational knowledge bases developed in these processes are hard to maintain and to keep updated.

This thesis discusses two approaches which enable the individual to practice personal knowledge management on-the-job with small entry costs and little overhead. At the same time, both approaches include the possibility of sharing knowledge resources with others, so that knowledge management activities can be performed in a collaborative fashion without centralized control or planning. We call this *self-organized collaborative knowledge management*.

The first part of the thesis introduces different aspects of solutions for *ontology-based peer-to-peer knowledge management* (P2PKM). After the description of an application, the *Courseware Watchdog*, which can be used for the management of personal learning objects and (among others) for the sharing of semantic descriptions of learning objects in a peer-to-peer network, two problems and possible solutions are discussed which arise when building such a P2PKM application.

First we consider the problem of *query routing* within a P2PKM network, i. e., the forwarding of query messages to peers which are likely to be able to answer the respective query. A connected topic is the question of suitable network topologies for routing; we introduce one possibility of *self organizing* topologies that adapt such as to support query routing.

For routing and topology construction, we need a way for peers to assess a priori the contents of other peers, so that they can make appropriate routing decisions. We propose a strategy for the computation of compact self descriptions of knowledge bases (and thus of peers) that can be used for the self-organization of the network.

The second part of the thesis is concerned with another approach for lightweight knowledge management, the so-called *folksonomies*. While in the P2PKM approach formal, semantically rich descriptions of contents were exchanged, folksonomies aim at simplifying the annotation of resources as far as possible and thus at attracting large numbers of users. In folksonomies, resources such as web bookmarks are annotated by freely chosen keywords, so-called *tags*.

As they are very easy to use, folksonomy services have attracted large user communities within a very short time. Therefore, large knowledge bases are available which can be represented as hypergraphs.

We analyze the global structure of these hypergraphs with measures from social network analysis. As the folksonomy graphs have a particular, non-standard structure, appropriate measures have to be defined.

As folksonomies are growing, typical information retrieval problems have to be solved. As user queries can yield very large numbers of possible results, the results to be presented to the user need to be ranked and selected. We develop a ranking algorithm for folksonomies, which furthermore can be used to compute personalized rankings in accordance to user preferences.

A related problem is the detection of structures and dependencies within a folksonomy. We propose a method for the computation of so-called *association rules* in folksonomies, generalized to the different dimensions. Depending on the selection of input dimensions, we can compute user communities sharing common interests, subsumption hierarchies of tags, or groups of resources on a given topic. Together with the abovementioned ranking algorithm, these clusters can be extended to fuzzy clusters. The results of this rule mining can then be used, e. g., in the user interface of a folksonomy tool to provide recommendations.

Zusammenfassung

In der heutigen Zeit wird *Wissen* in vielen Organisationen als die wichtigste ökonomische Ressource betrachtet. Die Akquisition, Bewahrung, Entwicklung und Benutzung von Wissen ist daher eine wichtige Aufgabe. Solche Aktivitäten, die auf die Wissensbasis einer Organisation einwirken, werden unter dem Begriff *Wissensmanagement* zusammengefaßt.

Wissensmanagement wurde oft als eine Aufgabe verstanden, die – ähnlich wie im Software-Engineering – in strukturierten Entwicklungsprozessen eingeführt und geplant werden kann. Es zeigt sich allerdings, dass solche Wissensmanagementaktivitäten oft nicht den Bedarf nach der Verwaltung persönlicher Wissensressourcen jedes Einzelnen decken können; weiterhin sind die Wissensbasen, die in solchen Prozessen erstellt werden, oft schwierig zu pflegen und veralten schnell.

In dieser Arbeit werden daher zwei Ansätze vorgestellt, die es dem Einzelnen erlauben, persönliches Wissensmanagement mit kleinen Einstiegskosten und geringem Mehraufwand während seiner täglichen Arbeit zu betreiben. Gleichzeitig zielen beide Ansätze darauf ab, dass die Wissensressourcen jedes Einzelnen mit Anderen geteilt werden können, so dass Wissensmanagementaktivitäten in kollaborativer Weise gebündelt werden. Dieses findet ohne zentrale Steuerung oder Planung statt; wir sprechen daher von *selbstorganisiertem, kollaborativem Wissensmanagement*.

Im ersten Teil der Arbeit werden verschiedene Aspekte von Lösungen im Bereich des *ontologiebasierten Peer-to-Peer-Wissensmanagements* (P2PKM) vorgestellt. Nach der Beschreibung einer Applikation, des *Courseware Watchdog*, die zur Verwaltung von persönlichen Lernobjekten und (unter anderem) zum Austausch von semantischen Beschreibungen der Lernobjekte in einem Peer-to-Peer-Netzwerk dient, werden zwei Probleme und Lösungsansätze diskutiert, die im Rahmen einer P2PKM-Applikation zu lösen sind.

Zunächst wird auf das Problem des *Routing* von Anfragen in einem P2PKM-Netzwerk eingegangen, d. h. des Weiterleitens von Anfragen

zu geeigneten Peers, die möglicherweise eine Antwort auf die jeweilige Anfrage liefern können. Verbunden damit ist die Frage nach geeigneten Netzwerktopologien für das Routing; wir stellen eine Möglichkeit vor, wie sich eine vorteilhafte Topologie *selbst organisieren* kann.

Dazu wird eine Möglichkeit benötigt, wie Peers im P2P-Netzwerk den Inhalt anderer Peers a priori abschätzen können, um Routingentscheidungen zu treffen. Wir stellen eine Strategie zur Berechnung von kompakten Selbstbeschreibungen von Wissensbasen und damit von Peers vor, die für die Selbstorganisation des Netzes genutzt werden kann.

Der zweite Teil der Arbeit beschäftigt sich mit einem weiteren Ansatz für „leichtgewichtiges“ Wissensmanagement, den sog. *Folksonomies*. Während im ontologiebasierten P2PKM formale, reichhaltige Beschreibungen von Inhalten ausgetauscht werden, zielen Folksonomies auf die einfachstmögliche Annotation von Ressourcen durch große Benutzergruppen ab. Dabei werden Ressourcen, z. B. Web-Lesezeichen, durch frei gewählte Schlüsselwörter, sog. *Tags*, annotiert.

Durch die Einfachheit der Benutzung haben solche Dienste in kurzer Zeit große Benutzergemeinden erreicht. Dadurch werden sehr große Wissensbasen verfügbar, die als Hypergraph darstellbar sind. Wir analysieren die globale Struktur des so entstandenen Hypergraphen mit Maßen aus der Theorie sozialer Netzwerke; da es sich allerdings um eine spezielle Graphstruktur handelt, müssen zunächst geeignete Abwandlungen der Maße entwickelt werden.

Mit wachsender Größe stellen sich in Folksonomy-Systemen typische Probleme aus dem Bereich des Information Retrieval. Wenn zu einer Benutzeranfrage große Mengen an möglichen Resultaten präsentiert werden können, muss einen Bewertung und Auswahl der Resultate erfolgen, die dem Benutzer präsentiert werden sollen. Wir stellen einen Ranking-Algorithmus für Folksonomies vor, der dieses leistet und darüber hinaus personalisierte Rankings in Abhängigkeit von Benutzerpräferenzen erstellen kann.

Ein verwandtes Problem ist die Erkennung von Strukturen und Abhängigkeiten in einer Folksonomy. Wir stellen ein Verfahren vor, das die Berechnung von sog. *Assoziationsregeln* verallgemeinert auf die verschiedenen Dimensionen einer Folksonomy. Je nach der Auswahl der Eingabedimensionen lassen sich damit z. B. interessenspezifische Benutzergruppen, Subsumptionshierarchien zwischen Tags oder thematisch verwandte Ressourcen errechnen. In Verbindung mit dem o. g. Rankingalgorithmus lassen sich diese Strukturen zu unscharfen Clustern erweitern. Die so gefundenen Gruppen oder Hierarchien lassen sich zur Un-

terstützung des Benutzers einsetzen, z. B. um Vorschläge für die Wahl geeigneter Tags zu unterbreiten.

Overview of Author's Contribution

Much of the work presented in this thesis originated from discussions with colleagues, particularly in the Knowledge and Data Engineering (KDE) Group at the University of Kassel, but also in the Knowledge Management Group of the AIFB Institute at the University of Karlsruhe and the Research Center L3S at the University of Hanover. In order to clarify the original contributions of the author of this thesis, those parts which stem from collaborative work with others will be defined more clearly. All parts of the thesis which are not explicitly mentioned below are the sole work of the author.

Chapter 4 describes the Courseware Watchdog, a system that was conceived and implemented in the PADLR research project when the author was with the Knowledge Management Group at AIFB Karlsruhe. The work on the Watchdog was split between Julien Tane and the author. The author contributed the connection to the Edutella network and the focused crawler. Julien Tane contributed the ontology evolution, clustering, and visualization modules. The overall conception, design, architecture, and the integration efforts were carried out by both.

Section 8.1: The formal model of folksonomies resulted from numerous discussions in the KDE working group.

Section 11.2: The general idea of applying a PageRank-like weight spreading scheme to the ranking within folksonomies arose in discussions in the KDE group. The author contributed the differential ranking scheme that was finally used as well as the Matlab implementation of the algorithm, including code to handle the large-scale datasets, and conducted the experiments.

Section 11.3: The general idea of this section was conceived in discussions with other KDE group members. The author provided the implementation, visualization, and conducted the experiments.

1. Introduction

This thesis is entitled "Self-Organized Collaborative Knowledge Management". In this chapter, we will discuss the three aspects of the title, introduce the main goals of the thesis, provide an overview, and outline its contributions.

1.1. Self-Organized Collaborative Knowledge Management

In this thesis, we will discuss methods for enabling and facilitating *self-organized collaborative knowledge management*. Reading this title from the back, we see three main topics of the thesis:

Knowledge Management: The thesis is concerned with knowledge management, i. e., the organization of knowledge assets and knowledge creation processes.

Collaboration: The main focus in on these knowledge processes happening between individuals in a collaborative fashion.

Self-Organization: Our goal is to enhance the space in which collaborative knowledge creation and sharing takes place, such that knowledge processes can self-organize in a fashion that is most useful for all participants.

In the following sections, we will explain these three components in more detail and give a guide through the structure of this thesis and its contributions.

1.2. Knowledge Management

It is a truism nowadays to emphasize the dependency of organisations and individuals on knowledge in order to compete successfully and to work in a knowledge society; for example, the European Commission has set itself the goal to become "the most dynamic and competitive knowledge-based economy in the world" in its Lisbon Strategy (European Commission, 2004). Knowledge management (KM) has been recognized as an important task in order to maintain and improve an organisation's capabilities in the knowledge-driven business environment.

1.2.1. Definitions

The purpose of this thesis is not to give a comprehensive overview over the KM area as a whole and the different facets and viewpoints involved.

Still, in order to be able to talk about KM, we will need a definition of "knowledge" and "knowledge management" to work with. Probst et al. (2003) offer the following definitions (translated from German by the author):

> Knowledge is the entirety of skills and capabilities which is used by individuals to solve problems. This entails theoretic insights as well as practical every-day rules and instructions. Knowledge is supported by data and information, but other than these it is always tied to persons. It is constructed by individuals and represents their expectations about cause-and-effect relations.
>
> (Probst et al., 2003; p. 22)

> Knowledge management (KM) is an integrated intervention concept which is concerned with possibilities of shaping the organizational knowledge base.
>
> (Probst et al., 2003; p. 23)

These definitions provide the outline for the activities and approaches detailed in the rest of the thesis.

1.2.2. Knowledge Management Approaches

Process and Product Oriented KM Approaches

In different viewpoints on KM and the conceptual frameworks that exist to structure KM efforts, two classes of approaches can be distinguished (see (Abecker, 2004; Chapter 1) for a detailed discussion):

Product Oriented Approaches consider knowledge as a product that can be captured and manipulated like any other resource and thus be managed in the traditional sense.

Process Oriented Approaches emphasize the knowledge creation process between individuals. Knowledge as such is not considered "manageable"; the subject of management is the environment in which knowledge processes take place.

As will be argumented below, the focus of this thesis is less on the modeling and representation of knowledge than rather on the enabling and facilitation of knowledge exchange between individuals, thus we follow the latter approach.

Structured KM Methodologies

As in software or business process engineering, process models and methodologies have been established in order to guide knowledge processes and meta processes (Sure, 2003; Schreiber and de Hoog, 1999).

These methodologies prescribe process stages to be completed, tools to be used, and artifacts to be created to implement KM measures. The aspiration to a systematic, deterministic procedure can be exemplified in the following principles from the CommonKADS methodology (Schreiber and de Hoog, 1999; p. 15ff):

- "Knowledge engineering is not some kind of 'mining from the expert's head,' but consists of constructing different aspect models of human knowledge."
- "The knowledge-level principle: in knowledge modelling, first concentrate on the conceptual structure of knowledge, and leave the programming details for later."
- "Knowledge has a stable internal structure that is analyzable by distinguishing specific knowledge types and roles."

- "A knowledge project must be managed by learning from your experiences in a controlled 'spiral' way."

These rather strict KM processes disregard, however, that knowledge creation and sharing often occurs in unexpected ways and that the overhead of adhering to strict practices may keep individuals from sharing knowledge in the first place.

SECI and Ba

Rather different from the abovementioned view on KM methodologies as a set of deterministic, engineered processes, another school of thought questions the manageability in the abovementioned sense of KM related efforts:

> It is our strong conviction that knowledge cannot be managed, only enabled. Since the publication of *The Knowledge-Creating Company* by Nonaka and Takeuchi in 1995, the concept of knowledge as competitive advantage of a firm has been drawing considerable attention from the corporate world and management academics. [...] However, the term *management* implies control of processes that may be inherently uncontrollable or, at least, stifled by heavy-handed direction.
>
> From our perspective, managers need to support knowledge creation rather than control it [...]
>
> (Nonaka and Nishiguchi, 2001; p. vii)

Accordingly, Nonaka and Nishiguchi do not discuss detailed process models, implementation phases, or technologies to attain knowledge-related goals, but rather emphasize the importance of providing the right environment for knowledge creation.

In their works about enabling knowledge creation in companies (Nonaka and Takeuchi, 1995; von Krogh et al., 2000; Nonaka et al., 2000; Nonaka and Nishiguchi, 2001), they have outlined three major components of a successful knowledge creating process (see Figure 1.1):

The SECI Knowledge Conversion Process: In this process, knowledge is created, transferred, and expanded via successive conversions between *explicit* knowledge—knowledge that is codified in an external representation—and *tacit* knowledge—knowledge that is only present in the minds of individuals.

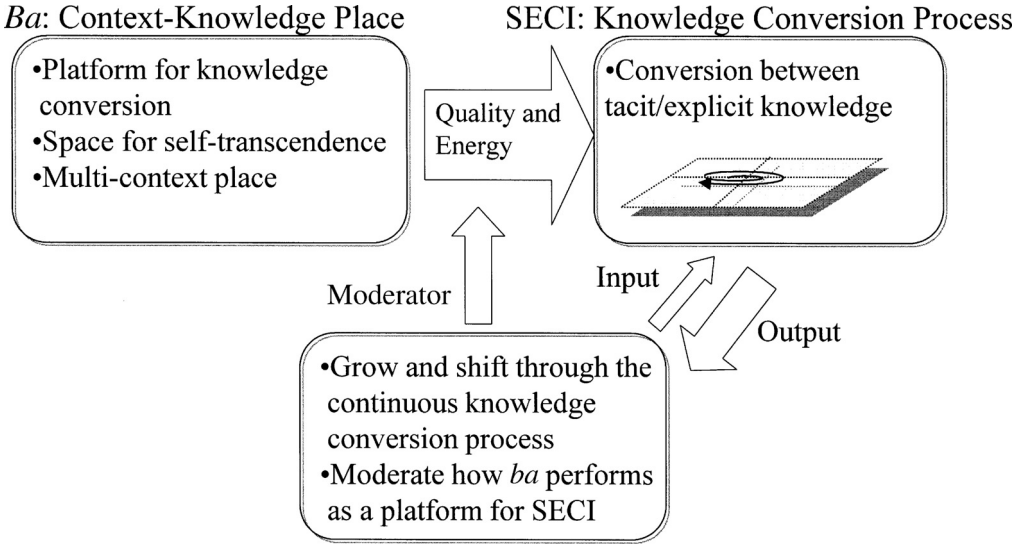

Figure 1.1.: Three elements of the knowledge-creating process (from (Nonaka et al., 2000))

This also entails the communication of knowledge between individuals, either by *socialisation*—sharing tacit knowledge through common experiences and face-to-face meetings—as well as by externalizing knowledge by one individual and subsequent internalisation by another.

Ba: The SECI process needs a space in which the knowledge conversions can take place. Nonaka and Takeuchi (1995) use the term *ba* to designate this environment. *Ba* is a Japanese word roughly meaning "space", but not only in the sense of place and time, but rather as "a concept that unifies physical space such as an office space, virtual space such as e-mail, and mental space such as shared ideals." (Nonaka et al., 2000)

Ba thus comprises, among others, the organizational, cultural, and technological dimensions of the place where knowledge creation and sharing between people takes place.

Moderator: While, in the opinion of Nonaka et al., knowledge can not be "managed" as such, a moderator—e. g. a manager in a company—can influence the knowledge creation process by shaping and participating in the *ba*.

In this thesis, the goal will be to provide systems and thus technical foundations for a *ba* in which knowledge sharing among individuals in the SECI process can take place efficiently and effectively.

Personal Knowledge Management

In addition to KM solutions that are implemented in an organization in a centralized fashion, there are always bits and pieces of personal knowledge captured, e. g., in the documents on a personal computer which will be managed by each individual knowledge worker on his own.

Tsui (2002) lists a variety of tools for personal KM that are used on a day-to-day basis by knowledge workers under the pressure of capturing knowledge on-the-job with a minimal overhead. Examples for these tools include Personal Information Managers (PIM), email applications, mind-mapping tools, or office documents with search capabilities on the user's personal computer.

One important question regarding the usefulness of KM tools is their integrability into each individual's personal workflow. Davenport (2006) stresses that "[i]n fact, 'integrating knowledge management into business processes' was selected as the most important issue of knowledge management in a 2002 survey", and further, "[w]hile there are several ways to bake knowledge into knowledge work, the most promising approach is to embed it into the technology that knowledge workers use to do their jobs".

In this thesis, we adopt this point of view and focus on two KM paradigms that emphasize on the individual's requirement of little overhead and little interruption of the personal workflow.

1.3. Collaborative KM Beyond Structured Processes

1.3.1. Collective Abilities and Informal Networks

From the discussion of the knowledge-creating process in Section 1.2.2, it is clear that knowledge management depends heavily on the collective abilities of many persons who share knowledge and expand and enrich it in the process. Probst et al. (2003; p. 20) state that "collective knowledge, which comprises more than the sum of the knowledge of a number of individuals, is of particular importance for the long-term survival of an organisation" (translated by the author). This collective

knowledge often turns out to develop outside the boundaries of corporate hierarchies and processes.

Davenport and Prusak (1998) consider the daily practice of knowledge workers as opposed to pre-determined organisational structures and processes. They use a metaphor of ”knowledge markets” (Davenport and Prusak, 1998; p. 25f), in which buyers, sellers, and brokers interact to distribute knowledge. Factors such as altruism, trust, reciprocity, and repute are presented as the pricing system within these markets. The knowledge markets often disregard formal organizational structures and are driven by pragmatic factors: ”Knowledge markets cluster around formal and informal networks, so providing information about these networks is a good way to make knowledge visible” and further (Davenport and Prusak, 1998; p. 38): ”What sounds like workplace gossip is often a knowledge network updating itself.” Krackhardt and Hanson (1993) support the observation that the actual working practice in organisations often circumvents the official hierarchies and reporting relationships and develops its own network structure.

1.3.2. The Cathedral and the Bazaar: The Knowledge Acquisition Bottleneck

Aside from the above-mentioned “unmanageability” of knowledge as claimed by Nonaka et al., there are further problems with the kind of knowledge artifacts that are produced in heavily structured knowledge processes. Wagner (2004), for example, points out that formalized knowledge representations as the outcome of knowledge processes such as described in (Sure, 2003) suffer from several difficulties:

> “*Narrow bandwidth.* The channels that exist to convert organizational knowledge from its source (either experts or documents, or transactions) are relatively narrow.
>
> *Acquisition latency.* The slow speed of acquisition is frequently accompanied by a delay between the time when knowledge (or the underlying data) is created and when the acquired knowledge becomes available to be shared.
>
> *Knowledge inaccuracy.* Experts make mistakes and so do data mining technologies (finding spurious relationships). Furthermore, maintenance can introduce inaccuracies or inconsistencies into previously correct knowledge bases.

> *Maintenance Trap.* As the knowledge in the knowledge base grows, so does the requirement for maintenance. Furthermore, previous updates that were made with insufficient care and foresight ('hacks') will accumulate and will render future maintenance increasingly more difficult [...]."

Problems such as these occurring when knowledge is to be elicited for use in a KM system have been called the *knowledge acquisition (KA) bottleneck* (Hayes-Roth et al., 1983). Similar arguments are made for the special case of ontology development by Hepp (2007), namely, that ontology construction is often too time-consuming, too costly, and does not reflect the end users' understanding of the domain; furthermore, he stresses that there are conflicts for knowledge workers splitting their efforts between contributing to ontology maintenance and doing their actual work.

Wagner draws a parallel between the situation in knowledge management and a dichotomy present in open-source software engineering. As pointed out by Raymond (1998), there are two basic models of organization for free software projects: in the Cathedral model, software is built by "carefully crafted by individual wizards or small bands of mages", who control the development process and release versions of the software to the public. On the other hand, in the Bazaar model, everybody is invited to contribute according to his own possibilities. Unfinished versions of the software systems are available to the public, so that debugging and testing can be supported by a large number of early users. Transferred to the KM setting, a bazaar-style development of KM tools and processes would mean that a participatory style of interaction would be encouraged in order to get as much user input as possible, even if it meant that users would have to deal with less-then-perfect responses from the system at times.

Another parallel which we see in software engineering is the shift from top-down, heavily structured methodologies such as the Rational Uniform Process (RUP) (Jacobson et al., 1999) to agile software development methodologies such as Extreme Programming (XP). The latter, for example, proposes *simplicity*, *rapid feedback* and *embracing change* as its fundamental principles (Beck, 2000). These principles dictate that functionality be added in incremental steps such that new code can be integrated and tested immediately and fed back into the development lifecycle. The development cycles are thus shortened to their absolute minimum, while heavyweight processes such as the RUP assume that

after thorough analysis and design phases, a more or less finished software system will be implemented at the end.

The emphasis in heavyweight processes on completing analysis and design prior to actually implementing a software system may lead to what has been called "Analysis Paralysis" and "Death by Planning" (Brown et al., 1998)—software systems that never get out of the analysis or design phases, because too much emphasis is put on getting every analysis and design decision right the first time before implementation commences. Thus, by the time a system would finally be implemented, it might have become obsolete already. In this thesis, we will thus focus on two KM paradigms that encourage the participation of large numbers of users in a KM system by lowering the effort needed to contribute. This way, even if the quality of each individual contribution is lower than in a more structured KM endeavour, the chance that a "right" answer for a user's knowledge need will be present in the system can be increased.

1.4. Self-Organization and Knowledge Management

In order to minimize the burden for each individual, and to allow for the unforeseen knowledge exchanges and informal networks described above, we will focus on self-organized KM in this thesis. Self-organization has been the research topic for efforts from many different disciplines, including physics, social sciences, cybernetics, and many others. The goal of this thesis is to build useful knowledge management solutions in a fashion that relies on the self-organization capabilities of the KM system.

While the core topic of this thesis is not the science of self-organization *per se*, we will briefly introduce the main ideas, the nomenclature, and some related work in order to provide the necessary background in the area. According to Heylighen (2001), self-organization is "the appearance of structure or pattern without an external agent imposing it." This term thus describes the possibility of a system of interacting entities (molecules in a liquid, animals in a swarm, humans in a crowd), under certain conditions, to turn into a state which exhibits order, without any external agent forcing the system to do so. In the following, we will briefly introduce the main nomenclature for use in later chapters of this thesis, following Lucas (2002) and Heylighen (2001), relate self-organization to KM, and point out its role as an overarching paradigm for this thesis.

System: A number of interacting entities surrounded by a distinguishable boundary.

State Space: The set of all possible combinations of states of its constituent parts available to the system.

Fitness Landscape: Each state in the state space can be mapped to a value denoting its fitness as measured by a given fitness function. Plotting this fitness value in an additional dimension yields a fitness landscape, in which the fittest and least fit states are maxima and minima, respectively.

Attractor: An attractor is a local maximum of the fitness landscape, i. e., a point to which the system may converge if it follows the gradient towards higher fitness values in a so-called *adaptive walk*.

Basin of Attraction: A basin of attraction is the set of states surrounding an attractor which will converge to that particular attractor.

In this terminology, self-organization means that the system moves through its state space, until it finally converges to an attractor that exhibits a higher amount of order than the state it started from.

In this thesis, we will follow the process-oriented view on KM and propose two kinds of systems that can enhance the possibilities of knowledge workers to share and create knowledge in a collaborative fashion. The goal is to minimize the overhead and restrictions that are imposed on the individual when he or she tries to capture, share, or obtain knowledge, i. e., "lowering the knowledge transaction costs" (Prusak and Weiss, 2006). In this sense, the goal is to provide and shape an appropriate *Ba* that supports the knowledge creation and sharing process as much as possible. The two approaches we will examine—semantic P2P systems and folksonomies—each provide the technological foundation for a space in which knowledge sharing can take place.

Put another way in self-organization terminology, we aim to create environments for self-organized KM systems that have attractors with the largest possible basins and the highest fitness values regarding usefulness for the users. We will try to achieve this, e. g., by implanting appropriate local behavior into peers or by supporting folksonomy users in their tagging behavior. Furthermore, we will observe KM systems through their evolution and analyze their way through the state space.

1.5. Contributions of this Thesis

In order to build such environments in which lightweight, collaborative knowledge creation and sharing can take place, we will address the following research questions.

Peer-to-peer Knowledge Management

1. *What can end-user applications for semantic P2PKM look like?*

   We will demonstrate in a prototype from the e-learning domain what an end-user application to be used in a P2PKM network could look like. It incorporates a number of modules which contribute to a unified knowledge base of the user's learning objects, which can be shared in a P2P network.

2. *How can semantic topologies in P2PKM networks organize themselves?*

   In order to facilitate routing of messages and discovery of peers with similar interests, we employ greedy network rewiring strategies that optimize the network topology without any centralized control.

3. *How can peers be described automatically to facilitate routing and the self-organization of the topology?*

   We provide an algorithm that allows for the concise representation of knowledge bases, such that peers can describe themselves semantically. These self-descriptions can then be used to make routing and rewiring decisions.

Folksonomies

1. *What are the structural properties of folksonomies?*

   We provide a formal model of folksonomies, and introduce measures for their global structural properties. According to these measures, the evolution of two large-scale folksonomy datasets is examined over the course of more than one year each.

2. *How can users be supported when searching and browsing folksonomies?*

   For the presentation of folksonomy contents to the user, a ranking of results is needed in order to present the relevant resources most prominently. We develop a personalized ranking algorithm

for folksonomies that can also be used to trace the connections between the dimensions of a folksonomy, e. g., by highlighting the most relevant resources for a given set of tags or vice versa.

3. *Can additional, useful information be mined from folksonomies?*

   In order to support the user in browsing the folksonomy and to generate useful recommendations, we will reduce the folksonomy to two dimensions using various projections, and mine the structure of the folksonomy to extract association rules. From these rules, recommendations can be generated and communities can be extracted.

   We also show how the crisp, rule-based results from this mining step can be extended to fuzzy sets of topically related tags, users, or resources with the abovementioned ranking algorithm.

1.6. Structure of this Thesis

As this thesis is concerned with two different approaches for self-organized collaborative knowledge management, namely, peer-to-peer knowledge management and folksonomies, Part I of the thesis covers peer-to-peer knowledge management, while Part II is concerned with folksonomies.

Chapters 2 and 3 start the first part with an introduction to P2PKM and the Semantic Web. In Chapter 4, we describe the Courseware Watchdog, a tool for P2PKM based on Semantic Web technology. The following chapters discuss two particular issues that need to be solved in a P2PKM environment: Chapter 5 deals with the problem of *query routing* in a semantic P2P system. Another building block for semantic topologies is introduced in Chapter 6, namely, the provision of concise self-descriptions of peers in a semantic P2PKM system.

The second part of the thesis discusses folksonomies as a second approach for lightweight, collaborative KM. After an introduction to folksonomies in Chapter 7, we present a formal model for folksonomies in Chapter 8. Chapter 9 gives a detailed description of the the datasets we will use throughout Part II. In Chapter 10, we examine the small-world properties of folksonomies, giving an overview over the structure of the folksonomies as a whole. Two algorithms for mining and ranking on folksonomies are discussed in Chapter 11, as well as the combination of both. Chapter 12 concludes the thesis with an outlook on future work and research directions.

Part I.

Peer-to-Peer Knowledge Management

2. Motivation, Use Cases, and Existing Systems

In this chapter, we will motivate why the combination of peer-to-peer technology and knowledge management efforts is a promising one. We will showcase some existing systems and describe use cases in which KM in P2P systems can yield a benefit over centralized solutions.

2.1. Motivation

"The story is clear: the internet was designed with peer-to-peer applications in mind, [...]" (Oram, 2001; p. 18) In his seminal book, Oram argues that the very structure of the internet is targeted at a peer-to-peer interaction pattern.

In the same volume, Shirky (2001) offers two definitions of the term "Peer-to-Peer":

> Peer-to-peer is a class of applications that takes advantage of resources—storage, cycles, content, human presence—available on the edge of the Internet.
>
> [...]
>
> So if you're looking for a litmus test for peer-to-peer, this is it:
>
> 1. Does is allow for variable connectivity and temporary network addresses?
> 2. Does it give the nodes at the edges of the network significant autonomy?

Although the World Wide Web would not be considered to be a P2P system according to these definitions, it was originally conceived much more as a collaboration platform (Berners-Lee, 1999), in which users act as information providers and consumers at the same time, than as the client-server communication platform is has been during the first decade of its existence. At that time, the web was largely dominated by relatively few large web sites such as Amazon or Google, which are consumed by many individual users. Only recently, with the "rise of social software" (Tepper, 2003), a person-to-person kind of communication is being re-established in the shape of blogs, wikis, and P2P applications.

Furthermore, the Semantic Web as an extension of the web in which available content is given a meaning was designed with the idea in mind that information can be gleaned from various sources in a piecemeal fashion and integrated (Berners-Lee et al., 2001)—a paradigm which, though the original paper speaks of "agents", is ideally suited for a P2P application.

2.1.1. Peer-to-Peer and the Semantic Web

Against this background, Stuckenschmidt et al. (2006) argue that the Semantic Web and P2P will make an ideal combination of technologies. On the one hand, a sophisticated knowledge management system in a centralized fashion takes a considerable amount of effort to setup and maintain; thus, a certain investment must be made before the first benefits can be reaped. On the other hand, P2P systems run on users' machines can provide immediate rewards by making resources on other users' desktops available for everyone, such that, for example, duplicate work yielding redundant results can be eliminated.

Still, without sophisticated knowledge representation mechanisms, such as those available in the Semantic Web, these systems will only be able to support a limited set of operations, e. g. keyword searches, so that the resources which other users are willing to share cannot be exploited to their full potential. Thus, the combination of Semantic Web and P2P technology opens up a feasible way of combining rich knowledge representation formalisms on the one hand with low overhead and immediate benefit on the other hand. In the following, we will explore different use cases which can benefit from a Semantic P2P infrastructure.

2.2. Use Cases

2.2.1. Knowledge Sharing within Communities of Interest

As an example of a community of interest, consider university lecturers in a particular domain, e. g., database management systems. Any one of these lecturers will have a considerable amount of learning objects on his personal computer, including lecture slides, exercises, and practice exams.

If a lecturer is willing to share his resources and associated metadata such as "exercise 4.3 is about the third normal form, which is treated in the fourth chapter of my lecture, for which there are slides in http://.../slides_dbms_4.ppt" on a peer-to-peer network, other lecturers will be able to pinpoint relevant material on a given topic and make use of these resources.

2.2.2. The Social Semantic Desktop

Another possible application of semantic P2P networks that is currently being researched in projects such as Nepomuk[1] is the *Social Semantic Desktop*. Every user of desktop applications is faced with the problem that the office documents, emails, bookmarks, contacts, and many other pieces of information on their computers are being processed by isolated applications which do not maintain the semantic connections and metadata that apply to the resources.

For example, an email pertaining to a particular appointment, having attached a spreadsheet with relevant information, will usually be split into three disconnected parts upon being received: the appointment goes into the calendar, the mail text stays in the mail system, and the spreadsheet file is saved into some folder in the file system.

Semantic desktop applications such as Haystack (Quan and Karger, 2004) or Gnowsis (Sauermann et al., 2006) are aiming at a solution of this kind of problems by employing Semantic Web technology to represent metadata about resources dealt with in desktop applications, so that explicit semantic relationships between units of information can be retained.

Connecting these semantic desktops by a P2P network is the idea put forward by Decker and Frank (2004). They call the result the *Networked Semantic Desktop* or *Social Semantic Desktop* (the latter name being the one

[1] http://nepomuk.semanticdesktop.org/

most commonly used today). This idea has received a lot of attention lately and has spawned a successful series of workshops (Decker et al., 2005, 2006).

2.2.3. Large-Scale Knowledge Sharing about People: Social Networks

On-line social networks such as Orkut[2], LinkedIn[3], or Facebook[4] are attracting large numbers of users who are willing to describe themselves in user profiles in order to discover and be discovered by friends or potential new acquaintances. Similarly, Friend-of-a-Friend (FOAF) profiles[5] stored on conventional web servers are seeing widespread use by people who want to provide machine-interpretable descriptions of themselves in order to weave a social network with their friends, colleagues, and relations.

As these user profiles are inherently connected to a particular person, the logical step would be to have the profile of each user on the user's personal laptop or portable device, so that the user can take "his" peer with him, keep the profile up to date, and have control over who is or is not allowed to peruse his profile, instead of uploading it to one or more central servers.

2.3. Existing Systems

In the following, we will briefly review existing semantic P2P systems which have been developed recently for a variety of use cases similar to those detailed in the previous sections.

2.3.1. Edutella and the Courseware Watchdog

The goal of the PADLR[6] project was to "produce a distributed learning web infrastructure, which will facilitate greater flexibility and functionality at all levels of university teaching. This will enable knowledge and

[2] http://www.orkut.com/
[3] http://www.linkedin.com/
[4] http://www.facebook.com/
[5] http://www.foaf-project.org/
[6] Personalized Access to Distributed Learning Repositories

learning materials to be constantly restructured and remodeled, and so [they] can be individually accessed as and when they are needed".[7]

Towards this goal, we have developed the Courseware Watchdog application, which is discussed in more detail in Chapter 4. Using the watchdog, teachers as well as learners can model learning resources using Semantic Web knowledge representations and share these metadata over a P2P network called *Edutella* (Nejdl et al., 2002). The Courseware Watchdog in connection with Edutella thus supports use cases such as the first one from Section 2.2.1.

2.3.2. Bibster

Another semantic P2P application targeted at a particular use case is *Bibster* (Broekstra et al., 2004), developed in the SWAP[8] project. In Bibster, bibliographic records of publications are represented in RDF and shared over a semantic P2P network. Other than the Courseware Watchdog, Bibster and its user interface are restricted to a publication sharing use case.

In addition to a special user interface providing access to standard bibliographic metadata such as author, title, etc., Bibster provides the possibility of classifying publications according to an ontology and aims at improving the user's experience by employing techniques for ontology evolution, so that each user can streamline his or her classification system.

2.3.3. Conzilla/SHAME

Conzilla (Nilsson and Palmér, 1999) is dubbed a *concept browser* by its creators. Mainly, it provides a graphical interface for drawing *context maps*—graphical knowledge representations. As the Conzilla data representation is done in RDF, its backend can be connected to semantic P2P networks, in this case Edutella, as well, yielding a platform in which end users can exchange their graphical knowledge representation and even draw queries against the network graphically. The focus of Conzilla and SHAME, however, is on modeling knowledge and on customizable user interfaces rather than on being a P2P client.

A related application from the same group, SHAME (Naeve et al., 2005), also connects to the Edutella network. Other than Conzilla, it

[7] PADLR project homepage; http://www.l3s.de/english/projects/padlr.html
[8] Semantic Web and Peer-to-Peer; http://swap.semanticweb.org/

is not using a graphical notation. Instead, it builds custom user interfaces from standard building blocks—text fields, lists, and the like—dynamically from RDF schemas.

2.3.4. Edamok

The Edamok (Bonifacio et al., 2004) system provides an infrastructure for distributed knowledge management. One of the key focuses is the mediation and mapping between different knowledge representation schemes—so-called *contexts*—on users' peers. This is accomplished by reducing the context mapping to a satisfiability problem in propositional logic, and using a SAT solver to compute mappings. Other than the other tools introduced here, Edamok does not use RDF to represent knowledge, but a proprietary, XML-based format for expressing contexts.

2.3.5. DBin

While the P2P parts of the aforementioned systems follow a similar approach—peers maintain knowledge bases and queries are forwarded between peers in order to find relevant information—the DBin system (Tummarello et al., 2006) uses a different approach. It is argued that peers should not have to answer queries unselfishly, possibly bearing a high load in order to serve other peers. Instead, peers join groups, and in those groups so-called GUEDs (Group URI Exposing Definitions) are used as a profile describing which parts of RDF graphs are of interest to that particular group.

Those parts of the knowledge base of each peer matching the GUED are then propagated within the group, so that the knowledge bases of each peer grow monotonously, finally replicating the group knowledge on each peer. All querying operations in DBin can thus be processed locally at each user's peer. DBin also includes a flexible user interface based on so-called *Brainlets*, which can contain interface descriptions as well as canned queries which users can pose in a query-by-example fashion.

2.4. Conclusion and Outlook

In this chapter, we have outlined the idea of P2P knowledge management, described use cases and some actual P2PKM implementations. In

the remainder of this part, we will look closer at some of the problems that need to be addressed when building P2PKM applications based on Semantic Web technology.

In order to introduce the relevant concepts and nomenclature, we will give a brief overview of the Semantic Web and related technologies in Chapter 3. Particularly, we will define our understanding of ontologies as a base technology for semantic P2P knowledge management, and show how ontologies can be used to measure the similarity of entities in the P2PKM system.

When semantically annotated resources are to be shared in a P2P network, one of the main issues that need to be addressed is how other peers may find these resources, for example, learning objects of a particular kind on a given topic. This issue can be decomposed into two problems. First, there must be a way of describing the contents of each peer in a way that can be used in the network to guide routing. This implies that short descriptions of peers' contents need to be found. Second, given this information, an appropriate way of structuring the network topology and routing query messages has to be found, such that contents can be retrieved with as little as possible network load.

First of all, however, we will address the need for an end-user application that allows for managing knowledge and sharing it on a P2P network. Users will have to be provided with a sufficiently easy-to-use application to participate in a semantic P2P network. In Chapter 4, we will introduce the P2PKM application that we have developed, namely, the Courseware Watchdog, in more detail. In this case, the application is targeted at an e-learning use case, but similar tools could be used in other use cases as well.

Afterwards, we will focus on the other two of the abovementioned problems within P2PKM applications. In Chapter 5, we will show how network topologies beneficial for query routing can self-organize in a P2PKM network. In Chapter 6, a method for computing so-called *expertises*, i. e., semantic self-descriptions of peers, is proposed. These expertises are necessary in order to allow peers to make routing decisions and rewire the network topology.

3. The Semantic Web

As the following chapters will be concerned with peer-to-peer knowledge management systems realized using Semantic Web technology, we will first clarify the relevant terminology about the Semantic Web, ontologies, and ontology-based metrics in this chapter.

3.1. Introduction

In order to build P2PKM applications, we will make use of technologies that are used in an extension of the current web named the Semantic Web. To quote from the original article coining the term:

> The Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users.
>
> (Berners-Lee et al., 2001)

The idea of the Semantic Web is thus to enrich the current web with machine-interpretable annotations. This enables automatic reasoning over distributed information, so that, e. g., software agents can infer useful new information and create value for the user. In order to implement this vision, several building blocks will be needed. There need to be languages for asserting statements about resources on the web, for describing the concepts and relationships within application domains, and for expressing rules.

While detailing the full spectrum of ideas and technologies subsumed under the label “Semantic Web” would be beyond the scope of this chapter, we will briefly introduce the main concepts as far as needed in the remainder of this thesis. For a full introduction to the Semantic Web, refer to Antoniou and van Harmelen (2004).

3.2. The Layer Cake

The various protocols and mechanisms that are necessary to build a Semantic Web in the aforementioned sense are usually displayed in a so-called *layer cake*, i. e., in a number of layers where each layer builds upon the services and abstractions provided by the ones below it.

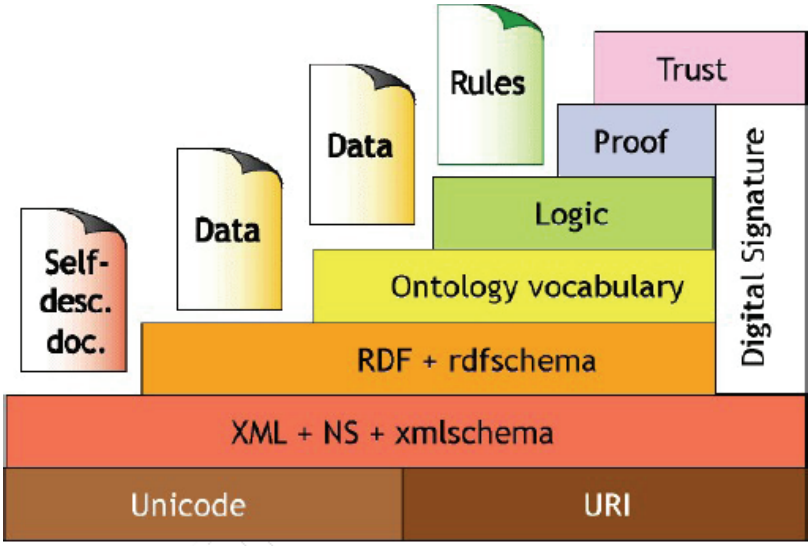

Figure 3.1.: The Semantic Web Layer Cake (taken from (Antoniou and van Harmelen, 2004), adapted from presentations by Tim Berners-Lee)

In the following, we will give a brief overview over the layers.

Unicode: Unicode is a character encoding standard which enables encoding characters in many different languages, including non-Latin ones such as Arabic, Hebrew, or Japanese.

URI: URIs (Uniform Resource Identifiers) (Berners-Lee et al., 2005) are identifiers for any kind of resource that should be described on the Semantic Web.

XML, Namespaces, and XML Schema: XML is the eXtensible Markup Language (Bray et al., 2006) which can be used to serialize tree-shaped data structures. It includes the concept of *namespaces*: namespaces serve to separate different elements that may otherwise clash due to their having the same name; by making the namespace a part of the element name, name clashes can be avoided. Namespaces

are identified by URIs. XML documents can be constrained by so-called Document Type Definitions (DTD) (Bray et al., 2006), which describe classes of valid XML documents for a particular application. Similar to DTDs, which are written in a language other than XML, XML Schemas (World Wide Web Consortium, 2004) allow to describe valid documents in XML and offer additional possibilities when describing data types.

Digital Signature: As the Semantic Web is about giving meaning to contents in order to enable automatic processing by agents, there has to be a mechanism for validating the authenticity of contents. This will be provided by digital signatures.

RDF and RDF Schema: RDF, the *Resource Description Framework*, is a language for describing resources on the web. Describing resources here means asserting statements of the shape *(subject, predicate, object)* about resources, wherein *subject* and *predicate* are resources themselves, and *object* can be either a resource or a literal value.

As an example, consider a representation of the following sentence: there is a thesis "Thesis 1" that was published at the University of Kassel, and that has the title "Self-Organized Collaborative KM".

In RDF, this would be expressed in the following statements:

Subject	**Predicate**	**Object**
uniks:thesis1	swrc:school	uniks:uni-kassel
uniks:thesis1	swrc:title	"Self-Organized Collaborative KM"

in which *uniks* and *swrc* are XML namespaces. Similar assertions can be added about the resource *uniks:uni-kassel*, and *uniks:thesis1* can also be the object of other statements. Thus, a set of statements makes up a labeled graph of RDF resources.

In addition to RDF, RDF Schema contains language elements for the description of *classes* and *properties*. RDF Schema coincides to a large extent with the notion of *ontologies* which we will introduce in Section 3.3, thus we skip the details here.

Ontology Vocabulary: The ontology layer of the Semantic Web layer cake consists of three different variants of the *Web Ontology Language (OWL)*. The three flavours of OWL, namely, *OWL Lite*, *OWL DL*, and *OWL Full*, offer different levels of expressivity, the first two corresponding to two different description logics (Baader et al.,

2003). They enable the description of concepts in terms of constraints, e. g., "father" could be expressed as "all instances of 'man' that are also in a 'has-child' relation to another instance".

Note that for the remainder of this thesis, we will restrict our usage of the term "ontology" to the definition of Section 3.3, which is more similar to the capabilities provided by the RDF Schema language and does not make use of advanced OWL constructs.

The remaining layers of the Semantic Web layer cake will not be needed in this thesis; we describe them briefly for the sake of completeness.

Logic: While the ontology layer can represent simple rules, e. g., the definition of "father" above could be read as "if X is a man and X has-child Y, then X is a father", the logic layer is intended to provide more powerful rule languages.

Proof: As the Semantic Web is about machine processable information that can be used through software agents, these software agents will have to exchange and validate proofs amongst each other such that they can prove that the results they deliver are correct. The proof layer will provide mechanisms for that.

Trust: Finally, the trust layer will consist of methods for expressing and evaluating to what extent another user or agent on the web can be trusted.

3.3. Ontologies

As this part is concerned with peer-to-peer knowledge management, a knowledge representation formalism to be used on peers is necessary. For the remainder of this part, we will assume that knowledge on peers is modeled in terms of an *ontology*. The definition of ontology used most often in the Semantic Web community is the one by Gruber (1993): "An ontology is an explicit specification of a conceptualization." Yet, this definition leaves a wide spectrum of possible interpretations regarding what an ontology actually comprises (Smith and Welty, 2001; McGuinness, 2003), of which the ontology layer in the layer cake as instantiated in the various dialects of OWL is one.

For the purpose of this thesis, however, we will consider a simpler form of ontologies in the sense that the KAON framework uses (Bozsak et al., 2002; Stumme, 2002b). The KAON model of ontologies is rather

close to RDF Schema, and does not entail the possibilities of the OWL family of languages in the ontology layer of the Semantic Web layer stack.

In short, a *core ontology* consists of a partially ordered set of *concepts*[1], the partial order being "subconcept of", and *relations*[1] between these concepts. For example, there could be concepts *Professor* and *PhDStudent*, and a relation *supervises(Professor, PhDStudent)* between them. A *knowledge base* or *OIModel* (for ontology-instance-model) consists of a core ontology plus instances of the concepts and relations; e. g. a knowledge base using the above concepts could contain *stumme* as an instance of *Professor*, *schmitz* as an instance of *PhDStudent*, and *supervises(stumme, schmitz)* instantiating the supervises relation.

3.4. Metrics on Ontology Entities

In the following chapters, we will need a way of assessing the similarity (or, conversely, the dissimilarity or distance) of entities within an ontology. An ontology of the kind described above can be viewed as a graph: the set of nodes comprises the entities, and the relations, relation instances, and the *subclassOf* and *instanceOf* relationships make up the set of edges. An edge between entities in this graph expresses relatedness in some sense: the relation *supervises(Professor, PhDStudent)* in the example above indicates that professors and PhD students have something to do with each other. On this kind of semantic structure, Rada et al. (1989) have proposed to use the distance in the graph-theoretic sense (lengths of shortest paths) as a semantic distance measure.

3.4.1. Metric Used in this Thesis

We follow the suggestion of Rada et al. and apply it to the aforementioned graph as follows:

- To each edge, a length is assigned; in order to account for the different kinds of relationships, taxonomic edges (instanceOf, subclassOf) get length 1, while non-taxonomic edges are assigned a length of 2. This reflects the fact that *subclassOf(PhDStudent, Person)* would be considered a closer link between these concepts than, say, *rides(Person, Bicycle)*.

[1] More precisely: *concept identifiers* and *relation identifiers*; we will stick to the simplified terminology of (Stumme, 2002b) here.

While Rada et al. (1989) state that non-taxonomic relations were of no value in their evaluations, it is also said that this depends on the actual use case. “person” and “bicycle” may not be considered similar although there exists a relation “rides” between them. On the other hand, topics such as “Specifying and Verifying and Reasoning about Programs” and “Software Program Verification”, which are connected by a “see also” relationship in the ACM Computing Classification System[2], might be considered very similar, so that we make use of the “see also” relation.

- Edge lengths are divided by the average distance of the incident nodes from the root concept. This reflects the intuition that high-level concepts such as *Person* and *Project* would be considered less similar than, e.g., *Graduate Student* and *Undergraduate* farther from the root, even if the graph distances between the respective pairs are the same.

- The lengths are normalized such that the longest distance in the graph equals 1.

3.4.2. Similarity, Relatedness, and Semantic Distances—Why Edge Counting?

Cognitive scientists, linguists, psychologists, and researchers in information science have long been exploring the notion of semantic similarity (things having similar features) and relatedness (things being associated with each other). Discussions about these and related phenomena and their respective properties have lasted for decades (cf. (Tversky, 1977; Gentner and Brem, 1999)). While most of this discussion is outside the scope of this thesis, some key points (Gentner and Brem, 1999) are worth mentioning: (a) Thematic relatedness and similarity are distinct phenomena. (b) Both can get mixed up or influence each other. (c) People mix relatedness or similarity in making judgments under different conditions.

In the context of this thesis, where the goal is to model knowledge bases in a P2PKM system, some more influences on the choice of the semantic distance are noteworthy:

- The ontologies to be used in a P2PKM will be engineered for KM purposes: the concepts, instances and relations between them will

[2] http://www.acm.org/class/1998/

have been included into the ontology for a reason. Thus, regarding a relation between two concepts as an indication that these two have something to do with each other reflects the intention of a knowledge engineer who has modeled that relation to express relatedness.

- In a P2PKM system, domain specific ontologies will be used. These represent a conceptualization of a small part of the world which is relevant for the given domain, so that arbitrary and possibly misleading chains of association (lamp—round glowing object—moon—etc.), which might be derived from a "world ontology", will not occur.

- Because a P2PKM system should be applicable in a wide range of situations, with different kinds of ontologies, modeling idiosyncrasies such as described in the next section need to be anticipated. This can be done by allowing for flexible weighting and filtering strategies.

While other approaches for metrics on conceptual structures such as ontologies have been proposed, these have limitations or make assumptions which are not fulfilled in our particular application. Approaches such as (Resnik, 1995) or (Tversky, 1977) assume the presence of full text or detailed linguistic background knowledge on the ontology; others such as (Maedche and Staab, 2002) only use concepts and an instanceOf relationship, neglecting instances and non-taxonomic relationships at all; still, this approach is also based on counting distances as edges traversed in a taxonomy. In order to yield maximum flexibility and to make use of as much of the modeled content of different ontologies as possible, an edge counting approach was chosen for this thesis.

Keeping this discussion in mind, one needs to be aware of what kinds of similarity or relatedness should be expressed in modeling the ontology and parametrizing the metric.

3.4.3. Caveats and Pitfalls on Real-World Ontologies

While the abovementioned edge-counting strategy of deriving metrics from semantic structures seems straightforward, applying it to ontologies used in real-world applications can turn out to be non-trivial:

Noise and Technical Artifacts: Not all of the content of a knowledge base may be genuinely taking part in the conceptualization of a certain

domain; e. g., in KAON lexical information is represented as first-class entities in the knowledge base. This leads to a large number of entities which are not relevant for the semantic distance computation. Similarly, there may a root class which every entity is an instance of, which would render our approach to calculating distances useless.

Modeling Idiosyncrasies: Engineering an ontology implies making design decisions, e. g. whether to model something as an instance or as a concept (Welty and Ferrucci, 1994). These decisions carry implications for the weighting of edges, e. g. if a taxonomic relationship is expressed by a special relation which is not one of *instanceOf*, *subclassOf*.

Implicit Edges: Some formalisms for modeling ontologies, particularly those allowing for richer, logic-based conceptualizations, enable implicit relationships that are inferred by a reasoner; a simple example would be that of a transitive relationship such as *ancestorOf* between persons. In that case, not all edges that may be worth considering will be present explicitly in the ontology and must be inferred. The ontologies as considered in this thesis do not offer these possibilities, however.

To overcome these problems, we have implemented extensive entity filtering and weighting customization strategies which are applied prior to the metric computation itself.

3.4.4. Obtaining of Proper Parameters

One question is how to choose the parameters, weighting schemes and filtering rules necessary for this kind of metric. Partly, these can be agreed upon just like the ontology to be used itself. When stakeholders agree that there should be a “see also” relation between topics, they could also agree on its importance or non-importance for retrieval tasks (cf. the discussion about the value of non-taxonomic relations in (Rada et al., 1989)). Meta-ontologies have been proposed to capture mapping information between ontologies (de Bruijn et al., 2004). Similarly, a meta-ontology can be used to contain this kind of information about metrics.

Secondly, one needs to note that this kind of semantic metric will not primarily be used to reflect human judgment of similarity or relatedness directly, but to structure a network topology so that queries can be

routed efficiently. For this type of use, optimal parameters can be determined in simulation experiments or might be learned over the lifetime of the system.

4. The Courseware Watchdog: A P2PKM Application

In this chapter, we describe the Courseware Watchdog, a prototypical P2PKM application which we have developed for an e-learning use case. With the Courseware Watchdog, users can maintain a knowledge base with metadata about their learning objects, e. g., lecture slides or exercises.
Using the Courseware Watchdog, the user can extend his knowledge base by crawling the web in a focused fashion and extracting new ontology entities from the crawled full text, or by participating in the Edutella P2P network, in which RDF descriptions about learning resources can be exchanged.
The work in this chapter has been published in (Schmitz et al., 2002; Tane et al., 2004).

4.1. Introduction

In recent years, mobile technology and Internet access have improved in such a way that it is reasonable to assume ubiquitous network connectivity and to rely on access to remotely stored content.

In the e-learning domain, the use of notebooks and mobile devices implies a new way of managing resources. The goal is, therefore, to exploit these chances as far as possible. Maurer and Sapper (2001), for example, predict an increasing role of mobile devices in education. The authors argue that e-learning has to be seen as a part of the general framework of knowledge management. To achieve this, it is important to integrate the technologies of these two domains.

Ontologies are regarded as one important foundation for KM activities in e-learning. In the E-Learning domain, standards have been devel-

oped to help describe learning objects.[1] Although these developments are a good start, there is a need for a more comprehensive approach which integrates the content, structure and evolution of the learning material. We present here our methodology and implementation of an ontology-based Courseware Watchdog, which supports the user in finding and organizing distributed courseware resources by offering a common framework for the retrieval and organization of courseware material. We illustrate this with a usage scenario.

In the first section we briefly expose the role of ontologies for e-learning. The following section discusses a prototypical usage scenario, from which we then derive the requirements for an ontology-based system for courseware management. This will lead us to an overall presentation of the integrated architecture of our Courseware Watchdog. In the four subsequent sections, we describe the specific modules of the Courseware Watchdog in more detail. Finally, we will sum up our results and discuss further research issues.

4.2. E-Learning in the Semantic Web

On a personal computer, it is possible to organize resources according to personal needs. In the case of remote resources, this is not possible anymore, since their storage is not under the control of the user. Through the use of hypertext, remote material can be linked and retrieved when needed. But the particular problem of finding and organizing this remote material becomes even more crucial.

Brase and Nejdl (2003) argue that standards like LOM (Nilsson et al., 2003) and Dublin Core [2] are gaining importance within the e-learning domain. They provide rich information on the learning material that is to be found in the web. However, their simple structure prohibits their use for modeling more complex knowledge. Stojanovic et al. (2001) explain how Semantic Web technologies based on ontologies can improve different aspects of the management of E-Learning resources. Indeed, ontologies are a means of specifying the concepts and their relationships in a particular domain of interest. Web Ontology languages, like OWL, are specially designed to facilitate the sharing of knowledge between actors (Staab et al., 2001) in a distributed environment. We wish to emphasize here on various advantages of using ontology-based metadata of learning resources.

[1] For more information on these standards, see http://ltsc.ieee.org/index.htm.
[2] http://dublincore.org/

From the modeling point of view, ontology languages are not only able to integrate LOM[3] and Dublin Core metadata, but also allow for the extension of the description of the learning objects with non-standard metadata, thus giving users and groups of users more flexibility when sharing resources.

Research in the E-Learning domain shows that standards are needed for interoperability,[4] but true interoperability does not only need data integration, it also has to consider the integration of applications. In this chapter, we illustrate such an integration of e-learning related applications with the implementation of a Courseware Watchdog. In the next section, we present a prototypical use case where such an integration is needed, from which we derive a set of requirements.

4.3. Use Case and User Requirements

To illustrate the purpose of our tool, we present a prototypical scenario in this section. It will show the different tasks that need to be addressed when trying to find and organize courseware material. While we use a teacher as an example for our scenario, similar points could be made for the case of a learner who wants to manage and share her portfolio of learning resources.

4.3.1. Usage Scenarios

Professor Meyer is a university professor at a German university in the domain of computer science. His main fields of activity are Data Mining and Knowledge Management. Since these fields of studies evolve very rapidly he has to be aware of the latest developments in these domains. At the beginning of the semester break, Professor Meyer prepares two lectures and two seminars which he will give during the next semester: the lecture “Introduction to Computer Science” designed for freshmen, a “Knowledge Discovery” lecture for more advanced students, a seminar on “Knowledge Management”, and a seminar on peer-to-peer and web services. He already has material from previous lectures but he feels that there is still room for improvement.

Professor Meyer expects to find a lot of material accessible on the web for the first lecture and he already has some lecture notes which are more

[3] See http://www.imsglobal.org/metadata/ on RDF(S) LOM Binding.
[4] See for instance the efforts of the Learning Technology Standard Committee.

or less ready to be used. He has already annotated part of the material with educational markup using the LOM and Dublin Core standards, and for the domain of computer science he uses the well known ACM taxonomy[5].

Browsing Existing Content

Professor Meyer planned the construction of the two courses in the following way: first, he needs to have a systematic overview over the material that he has collected by and by. Instead of using a generic ontology like the ACM classification, he may also use an ontology that was created by one of his colleagues. In that case, Professor Meyer needs to familiarize himself first with the content and the meaning of the concepts and relations, as the ontology may be created from a slightly different viewpoint.

Beside the most important concepts, the ontology also contains pointers to relevant resources that he collected (e. g., HTML pages, PDF files or powerpoint files). Some of the resources may be stored on the professor's computer, while others may be located at a remote location. The professor can access these resources by means of a standard browser.

Finding Relevant Material

Professor Meyer now wants to find new material. For this, he considers two approaches: either search for it in the world wide web or in distinct decentralized repositories that provide more structured semantic metadata about learning material.

Both tasks can again be supported by using an ontology. It provides means for extending search term combinations with semantically related concepts, and it serves as an interface to the (semi-)structured repositories.

Querying a Network of Semantically Annotated Material Suppose our professor has access to an Edutella[6] network (Nejdl, 2002). Edutella is a peer-to-peer (P2P) framework, in which the different peers provide semantically annotated metadata on learning material. It also allows for

[5] http://www.acm.org/class/1998

[6] In this scenario we will take Edutella as a prototypical distributed network of semantically annotated learning material. Another example is POOL, see (Hatala and Richards, 2002).

the integration of web services to gain access to material offered by libraries or similar repositories, see for example (Ahlborn et al., 2002).

In order to find new relevant material in the P2P network, Professor Meyer first needs to define a query. Professor Meyer searches for lectures on the topics "Algorithmics" or "Knowledge Discovery". Hence he defines the following query (denoted in a controlled language here which resembles RDF query languages such as QEL):

> Return every 'Lecture' which 'hasTopic' 'Algorithmics' or which 'hasTopic' 'Knowledge Discovery' and for each match retrieve also the values of the properties 'dc:title' and 'dc:author'.

He sends the query to the network and receives an answer. He can then access the retrieved resources by standard browsers and viewers. He can send more specific queries to the Edutella network to get further information about the specific lectures or authors that are of interest to him.

Finding Learning Material on the Web Professor Meyer knows some web sites that are relevant for his task. He is quite certain that some interesting material (or at least pointers to it) would be accessible there, had he only time to browse the sites and follow the hyperlinks.

One solution would be to apply a crawler that follows the links starting from these pages, and to collect the resources showing up. However, if every individual user started his own crawler, this would lead to an unnecessary overload of web traffic. On the other hand, Prof. Meyer is not interested in harvesting all pages accessible from the start URLs, but only in a specific subset. He selects a set of concepts from the ontology which specify the kind of pages he wants to retrieve. The crawler then scores each page and each hyperlink according to the frequency of these concepts on the whole page and around the hyperlink. Concepts that Meyer did not type in explicitly, but which are semantically related to these concepts within the ontology, also add to the score. The links with the highest score are followed next. This way, the structure of the ontology provides a complex measure of 'relatedness', which supports a focused crawling process, similar to the typical browsing behavior of a a human user.

He decides to send the crawler to search for new material on Knowledge Management and Peer-to-Peer. For this, he selects the corresponding concepts as well as other concepts that seem to be relevant in both

domains. Then he launches the crawler at the homepages of the European projects Ontoweb and SWAP which he considers as good starting points. The retrieved results can finally be browsed using the same ontology that was used to specify the preferences for the focussed crawler.

Clustering the Resources

Professor Meyer now wishes to organize the learning resources he has retrieved. Ideally, he wants to group similar documents together and structure the document collection according to certain criteria. He may use available clustering algorithms, but he does not want to ignore the additional background knowledge encoded within the ontology. For instance, he would expect that a lecture about 'Machine Learning' and another lecture about 'Data Mining' are grouped in the same cluster, even though they may not share many technical expressions. The ontology will then bridge the gap, as it subsumes both terms under 'Knowledge Discovery'.

Moreover, it is important for him to be able to understand the clusters. For this he needs specific visualization techniques, which allow him to understand why the documents have been grouped together. Here again, the ontology can be used to structure the presentation.

Ontology Evolution

Finally, Professor Meyer wants to be aware of the ongoing evolution of the vocabulary of his field. His ontology should reflect the changes and should also include upcoming topics. In the web, one can discover upcoming topics by new terms showing up within the retrieved documents, or by terms which existed before but now increase in frequency. The decision if a topic is relevant for Prof. Meyer, and if it has to be included in the ontology, is finally up to him, but seen the large number of upcoming terms, he would expect some tool support.

4.3.2. User Requirements

From the scenarios described in the previous section, one can derive several tasks that need to be supported by a Courseware Watchdog. These can be summed up as follows:

1. Understanding the ontology and browsing the content,
2. retrieving relevant material through focused crawling,

3. querying semantically annotated resource repositories,
4. clustering and organizing the documents according to the ontology,
5. updating the ontology and knowledge base according to current data.

The first task requires a user interface for accessing the ontology and the collected resources. The next two tasks deal with the acquisition of resources: querying P2P repositories and retrieving elements from the Web through crawling. These two tasks can be considered as complementary. Since in both cases the interaction with the ontology is necessary, they have to interact with the browsing interface. The same holds for the last two tasks. Here again, interaction with the user goes along the ontology.

Since ontologies are used in all these tasks, it is rather natural to use a representation of the ontology as a central part of the user interface. Whenever one of the other four tasks is performed, this ontology representation has to be complemented by a second interface specific to the task. By keeping the ontology a constant part of the interface, it serves as a mental fixed point and thus facilitates the orientation of the user.

Moreover, the scenario shows that there is a strong interaction between these five tasks: The crawling task needs the ontology as background knowledge, and populates the text corpus. The corpus will be structured by the crawling task, and serves itself as input to the ontology update task, which, in its turn, populates the ontology. The updated ontology can be used to launch again the focused crawler, or to retrieve resources in the P2P network. The step from one task to the next has to be made explicit in the user interface to support the user in keeping the orientation in the process.

In the rest of this chapter, we will describe how these requirements are turned into an architecture, and how this architecture is implemented within our Courseware Watchdog. The next section gives the overall picture of the Watchdog; sections 4.4.2 to 4.7 discuss the different parts in more detail. Each section provides first a conceptual view of the part, before discussing implementation details.

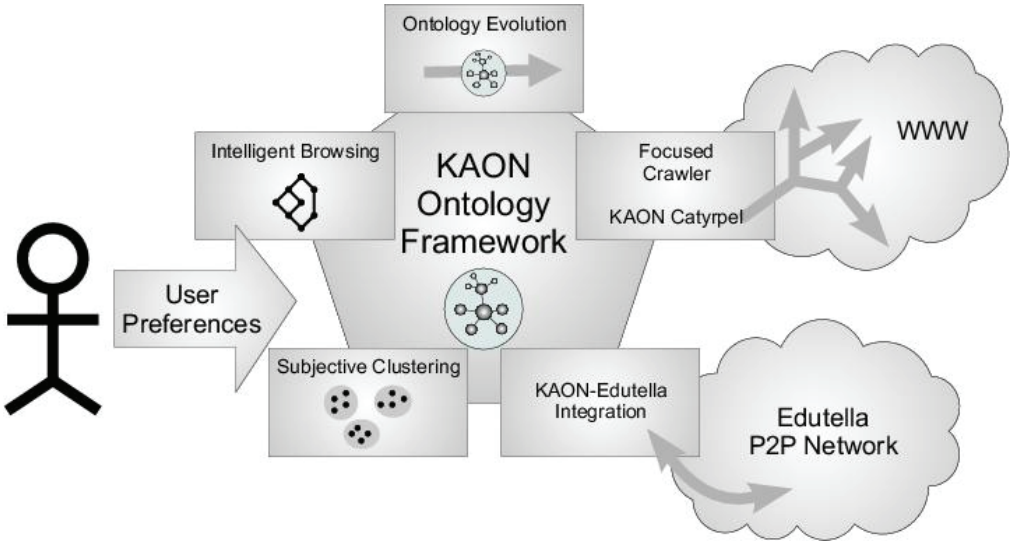

Figure 4.1.: The components of the Courseware Watchdog.

4.4. The Courseware Watchdog

The Courseware Watchdog[7] described in this chapter addresses the requirements mentioned above by using an approach which exploits concepts from the Semantic Web, such as ontologies, in an E-Learning scenario (Stojanovic et al., 2001). It is part of the PADLR project (Personalized Access to Distributed Learning Repositories) that builds upon a peer-to-peer approach for supporting personalized access to learning material[8].

4.4.1. Overview

When developing the Courseware Watchdog, we aimed at addressing the different problems evoked by the abovementioned scenarios. The tasks to be solved are addressed by different modules. One important goal was to use a single semantic model to integrate the different tools. We show that their combination offers the user a single simple tool for tasks depending on each other. The Courseware Watchdog consists of the following components which are organized around an ontology management system (see Figure 4.1):

[7] http://cwatchdog.sourceforge.net/
[8] http://www.l3s.de/english/projects/padlr.html

1. *Visualization and interactive browsing techniques* allow for the browsing of the ontology and knowledge base in order to improve the interaction between of the user with the content.

2. A *focused crawler* finds related web sites and documents that match the user's interests. The crawl can be focused by checking new documents against the user's preferences as specified in terms of the ontology.

3. An *Edutella peer* enables querying for metadata on learning objects with an expressive query language, and allows to publish local resources in the P2P network. Furthermore, the Courseware Watchdog can also access any repository offering metadata via an SQI web service interface (Simon et al., 2004).

4. A *subjective clustering component* generates subjective views onto the documents.

5. An *ontology evolution component* comprises ontology learning methods which discover changes and trends within the field of interest.

The components are not separated but there is a logical data flow between them, as discussed in the previous section. This is reflected in the architecture of the Watchdog, which is shown in Figure 4.2. The Watchdog is organized around the ontology and the text corpus. Section 4.4.2 discusses how these two are modeled using the KAON framework.

Both the ontology and the text corpus can be accessed by the user interface described in Section 4.5. A screenshot of the interface is shown in Figure 4.3. The left part of the screen always shows the ontology; it serves as a fixed point for the interaction of the user with the system. The right part of the screen changes, depending on which of the components crawler, Edutella, clustering, or evolution is currently active. These four components and their interaction with the other components are discussed in sections 4.6 and 4.7.

Implementation Details. The Courseware Watchdog application is built on top of the KAON Workbench (see Section 4.4.2). The workbench offers a plug-in interface for extension modules, which can make use of each other and the basic KAON abstractions.

All components of the Watchdog mentioned in this chapter are implemented as KAON modules.

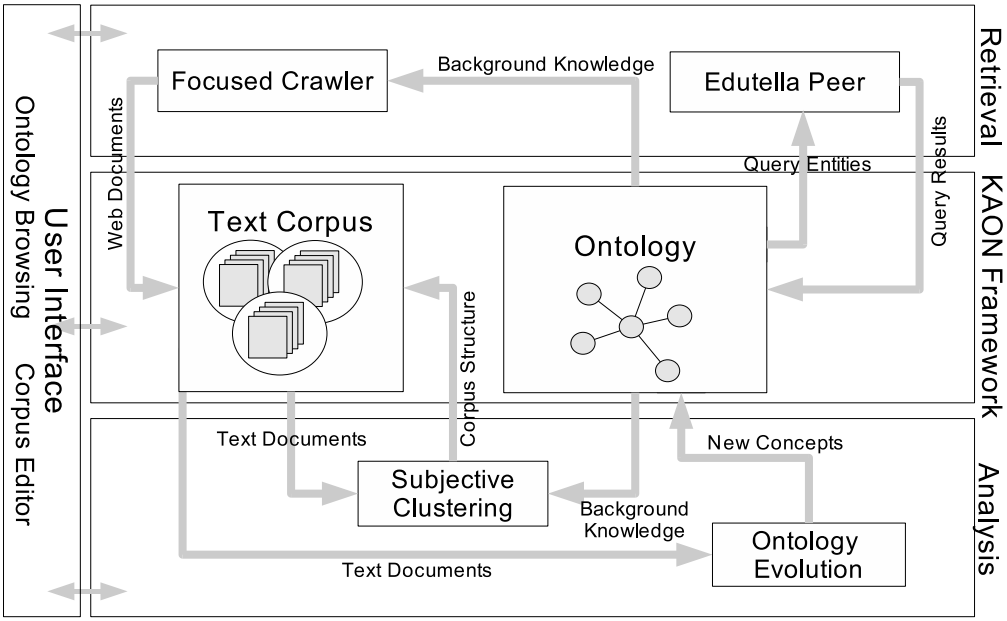

Figure 4.2.: The architecture of the Courseware Watchdog.

4.4.2. The Courseware Watchdog and the KAON Framework

All the modules use ontologies in the sense of Section 3.3; they are built on top of an ontology tool suite named KAON, the Karlsruhe Ontology and Semantic Web Framework (Bozsak et al., 2002). KAON offers abstractions for ontologies and text corpora, an ontology editor and application framework, inferencing, and persistence mechanisms, etc.

On this platform, integration in the Courseware Watchdog is achieved on several levels, namely:

- on the semantic level—through ontologies
- on the web structure level—the structure of the graph of web documents is stored in an ontology
- on the structure level of the corpus—the different algorithms for the clustering and ontology evolution use the same corpus model

This common integration model allows to use both the browsing and the querying of the resources available or discovered. Or it allows the interaction with the results of algorithm. For instance, it is then possible to use the clustering results as input to the ontology evolution.

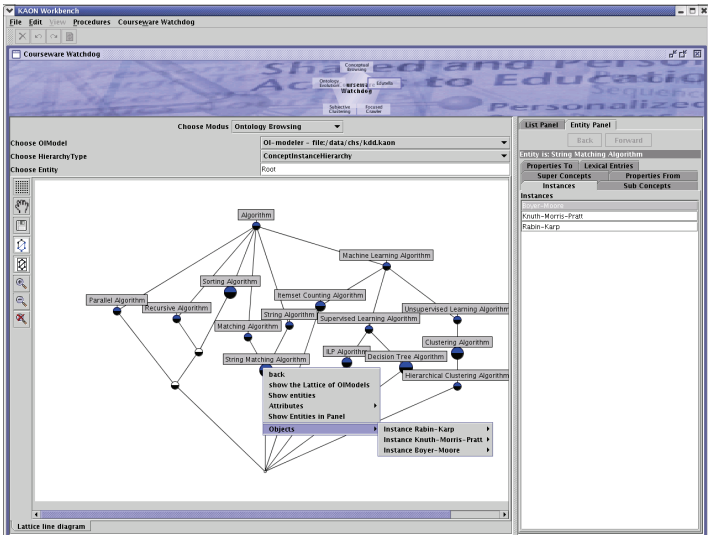

Figure 4.3.: The browsing interface of the Courseware Watchdog

4.5. The User Interface: Browsing the Watchdog Data

As shown in the scenario, the interaction of the user with the ontology is crucial for all ontology-based tools. In the Courseware Watchdog, this is done using the browsing component. In this section, we will first explain how the ontology is displayed. Then we describe how it can be used to interact with the ontology.

4.5.1. Displaying an Ontology

Ontologies are complex structures based two kinds of relations: *hierarchical* and *non-hierarchical* relations. For each kind of relation, we use an appropriate technique: the display of hierarchies through concept lattices in the first case, and relational browsing otherwise.

In order to display specific hierarchies,[9] we use techniques from Formal Concept Analysis (FCA) (Ganter and Wille, 1999; Stumme and Wille, 2000), a conceptual clustering technique which allows the display of hierarchies of concepts using lattices. We create the minimal lattice containing the hierarchy, the Dedekind-MacNeille completion (Ganter and Wille, 1999). This allows for the display of multiple inheritance. By adding instances, we get new concepts which also reflect eventual

[9]For example, we display concept, property or topic hierarchies.

multiple instantiation. Using these techniques, some new concepts may appear, even if they were not explicitly modelled within the ontology. For example, Figure 4.3 displays the hierarchy of subconcepts of “algorithm”. One can see that there are instances of “sorting algorithms” which are also “recursive” and “parallel” in the current knowledge base. This suggests that a new concept might be useful in this position (for details on ontology evolution, see Section 4.7.2).

This approach follows the idea of CEM, the Conceptual Email Manager (Cole and Stumme, 2000), which supports exactly such a kind of navigation in collections of emails. We extended this approach to the use of ontologies. When applied to learning material, multiple inheritance within this hierarchy provides a rich conceptual landscape for navigating and retrieving the educational media. For more details about the construction of concept lattices for ontology browsing, refer to (Tane, 2007).

Non-hierarchical relations existing in the ontology represent links between various elements of the ontology (e. g., the “lecturer” of a “course” should be linked by a relation “holdsCourse”). These kind of relation are best understood and used through some kind of exploration. Relational browsing is a technique offering the user different links which he can choose to follow, but in addition to normal browsing, the links are typed according to the ontology instead of the documents. It is possible to navigate and explore the ontology using the relations of the ontologies, and then display different kinds of hierarchies according to ones needs. By clicking on the concept “String matching algorithms”, one is, e. g., able to find its instances, such as the “Boyer-Moore” algorithm, or find the lectures which refer to this algorithm by following the relation “isReferredBy” of the instance “Boyer-Moore”(selected in the right panel of Figure 4.3.).

4.5.2. Interacting with the Ontology

Beside just displaying the ontology, the browsing component is also a generic way of interacting with the ontology. Therefore, this component plays a central role in the Courseware Watchdog. Indeed, we divided the main Watchdog panel in two parts. The left hand side always presents the browsing component, while the content of the right part depends on the actual usage.

We defined different modes, in order to simplify the interaction for the user. These modes correspond to the different tasks explained in

Section 4.3.2. In each mode, the user sees on the left hand side the lattice of some kind of relation. He may click on the nodes of the lattice to select them in order to reuse them later, in another context, or he may select one node, corresponding to a concept, and display more information on the right hand side. In the context of the evolution, he might introduce, as a subconcept of the selected concept, a new concept detected with the evolution module. In Figures 4.3, 4.4, and 4.5, you can see three different combinations corresponding to the browsing, the query refinement, and the annotation of clusters. In combination with these components, the browsing interface supports the construction of queries as well as the selection of entities which can then be used in the crawling, clustering or evolution process.

According to the context of use, the interface allows for the selection of entities and the application of certain actions on these. A typical example of interaction would be to look for documents containing information on certain topics. The user could then select the topics of interests from the topic hierarchies, and look for the document which are related to these by following the relation “isTopicOf”. Once the documents have been found, he can either open them, or insert them into a new or existing corpus.

Implementation Details The ontology browsing module uses the open source FCA software ConExp[10] as a library. It was extended to be able to browse ontologies. The creation of the data model uses various strategies to display only the necessary nodes in the lattices. Moreover, it was necessary to introduce dummy instances to the FCA software in order to maintain the structure of the concept lattice and the correctness of the ontology model at the same time. These dummy instances represent potential instances of the model for which there is no example present in the knowledge base. The ontology browsing view is used to interact with all other modules; thus, according to the current context, different actions are available in a context menu on ontology entities. For example, when using the retrieval components (see next section), the menu offers the possibility to fill in query variables with the current ontology entity.

[10] See http://www.sourceforge.net/projects/conexp.

4.6. Retrieval Components: Focused Crawler and Edutella

In this section, we will describe the two retrieval components which allow the user to find material according to his interests. In both cases, the ontology browsing component is used to define the elements that should be looked for. While the focused crawler uses user preferences to look for relevant terms in the full text of web documents, the Edutella network is used for exchanging metadata on learning resources which have been annotated semantically.

4.6.1. Focused Crawler

A web crawler is a program that collects data from the web automatically by following links extracted from web documents. Thus, a portion of the web is traversed in a breadth-first manner, usually without regarding the relevance of the collected documents with respect to the needs of one specific user.

The Courseware Watchdog includes a web crawler to retrieve learning material from the WWW. In order to restrict the traversal to material relevant to the user, the crawling process is *focused* (Chakrabarti et al., 1999). Focusing here means preferring those links in the crawling process that appear to be pointing to relevant documents.

The focused crawler of the Courseware Watchdog builds upon crawlers previously developed by the author and colleagues (Schmitz, 2001; Maedche et al., 2002a). It uses an ontology-based focusing strategy, relying on the KAON environment for ontology-based tools.

The user can specify his preferences by assigning weights to selected entities of the ontology. The preprocessing step of the crawler then uses the background knowledge present in the ontology—namely, relations between concepts or instances—to compute relevance scores for other entities.

For example, a user may specify that he is interested in material on “Machine Learning Algorithms”. The preprocessing step of the ontology will then also assign weight to an instance of that concept (e. g. “C4.5”), or to related concepts (e. g. “Algorithm”, being a superconcept). Various parameters of this spreading of weight can be manipulated: which kind of relations in the ontology to follow, how large a radius around a user-selected entity to consider, etc.

During the crawling process, terms in the web pages as a whole and in the anchor texts of hyperlinks in particular are then compared against the precomputed weight tables (after stop-word removal and stemming). Thus, scores for the web pages and for the hyperlinks on a page are computed.

This enables the use of different levels of "sharpness" in the focusing; e. g., one may decide to present only documents relevant according to a stricter weighting strategy to the user. On the other hand, the decision of which links to follow can be made based on a more forgiving weighting scheme, so that the crawler is able to "tunnel" over a page which is not relevant in itself, but may point to other relevant documents. This reduces the probability of getting stuck on one single non-relevant page.

The crawling process yields two kinds of results. First, the full text of the documents is stored in a text corpus. From there, it can be used in the clustering (see Section 4.7.1) and ontology evolution (Section 4.7.2) modules.

Second, the metadata of the crawling process—which pages have been crawled, which were the most relevant entities on a page, what is the link structure between pages—are stored in an ontology. From there, they can be presented in tabular form or using the ontology browsing component, or provided to other users using the Edutella module (see Section 4.6.2).

Implementation Details The crawler has a main-memory based infrastructure which is loosely modelled after Mercator (Heydon and Najork, 1999). To improve performance and at the same time limit the load on individual web servers, it assigns each page to one out of several crawl queues, each of which retrieves pages with a limited frequency, e. g. one page per minute.

The crawling strategy—in our case directed breath first search with ontology-based weighting as described above—is factored out in the source code as a distinct class, so that it is easily replaceable.

4.6.2. Integrating the Edutella Peer–to–Peer Network

The Courseware Watchdog includes the possibility to participate in the Edutella peer-to-peer network. The Edutella[11] network applies the P2P paradigm to exchange structured information about available learning

[11] http://edutella.jxta.org

resources (Nejdl et al., 2002). A common data model facilitates the integration of data sources such as relational database systems or XML and RDF repositories. Thus, all of these can act as Edutella peers.

The Edutella module allows the Courseware Watchdog to act in the Edutella network as a consumer as well as a provider of learning resource metadata. Other tools have been extended to act as Edutella peers as well, e. g. the Conzilla concept browser (Nilsson and Palmér, 1999), and can thus interoperate with the Watchdog.

Edutella Consumer

While Edutella peers support a powerful query language named *QEL* (Nilsson and Siberski, 2003), we chose to offer a simplified, easier-to-use interface to Edutella querying in the Courseware Watchdog (see Figure 4.4).

The user is given an extensible *query repository* of template queries, which he can submit with the given placeholders, or, at his choice, partially fill in with concrete values. This means that free variables of the query are replaced by literal values or entities (instances, concepts or properties) from the ontology. Thus, a query-by-example interface is provided which enables inexperienced users to pose meaningful queries, yet more experienced users can specify advanced queries as well.

Consider the following example, where capital letters indicate variables. The query

```
?-s(Entity, rdfs:type, Type),
  s(Entity, dc:title, Title).
```

(meaning: *Give me all resources that have a type and a title and return the respective types and titles*) can be made more specific by supplying entities to fill in variables (see figure 4.4).

The user might choose `lom:lecture` via the ontology browsing interface to replace `Type`, thus yielding the specialized query

```
?-s(Entity, rdfs:type, lom:lecture),
  s(Entity, dc:title, Title).
```

(Give me all lectures and their respective titles)

Writing QEL queries directly and storing them into the query repository for future reference is also possible for advanced users.

One other point worth mentioning is that QEL supports outer joins as well. In practice, this means that while a user can specify a number of

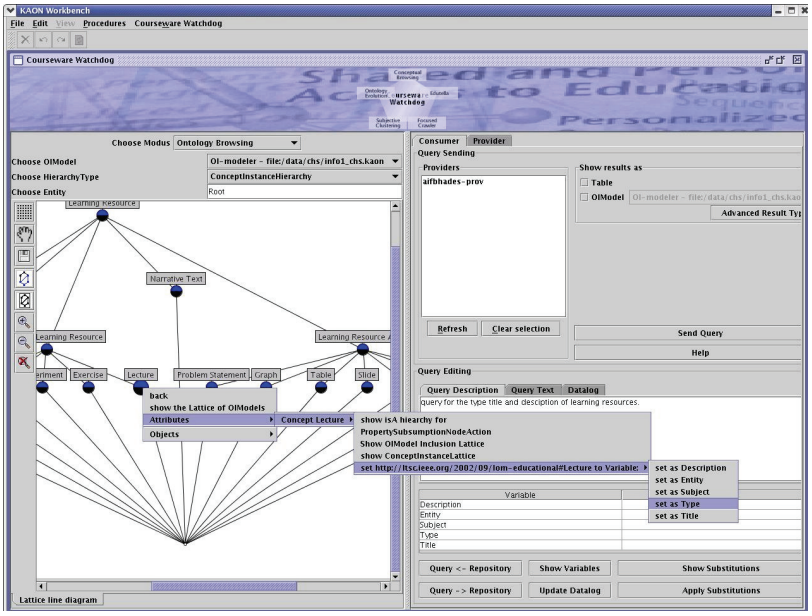

Figure 4.4.: Refining a query.

attributes or properties to be retrieved from the resources of interest, not all of those attributes need to be present on a particular resources for it to qualify as a query result, but if they are, they will be reported.

Edutella Provider

An instance of the Courseware Watchdog can also act as a provider of information in the Edutella network. At the users request, ontologies loaded into the workbench can be offered to the Edutella network, enabling other users to query this Watchdog instance for metadata.

Through use of the KAON ontology inclusion mechanism, the user retains control over what pieces of information he wants to provide to the public domain: if he decides to split his repository into a public part *Pub* and a private part *Priv*, he can still have a consistent view over his material by having *Priv* include *Pub*, and at the same time offer only *Pub* to other Edutella users.

SQI Repositories

The SQI protocol (Simon et al., 2004) has been established in the European Network of Excellence ProLearn[12] as the common denominator for querying structured repositories of learning resources. It is currently implemented by most present repositories as a SOAP-based webservice (Gudgin et al., 2006). As most of the present SQI repositories understand QEL as a query language, we also included the possibility of querying a known SQI endpoint via the same interface used for Edutella queries.

Implementation Details. Providing Edutella connectivity for the Watchdog posed two challenges. We had to cope with the mismatch of general RDF metadata in Edutella on the one hand versus KAON ontologies on the other hand. Furthermore, we needed to integrate two different RDF APIs.

First, the Edutella network deals with RDF metadata in general, while the Courseware Watchdog deals with KAON ontologies represented using a subset of RDFS. But as the Edutella community has agreed on a subset of LOM-RDF (Nilsson et al., 2003) which fits into the KAON ontology language, only minor adjustments were needed. These include the treatment of labels (which are dealt with differently in KAON and RDFS) and containers (Bag, Seq, etc.). Containers are not a part of the KAON language, but can be emulated using KAON constructs to provide a very similar functionality to RDFS containers. Should RDF statements outside the KAON ontology language be retrieved from Edutella, these are incorporated into the RDF model, but will not be visible in the ontology-based user interface.

Second, KAON comes with its own RDF parser and API, whereas Edutella relies heavily on the Jena RDF library[13]. Thus, we introduced an adapter which wraps the KAON RDF model representing the ontology behind a Jena API. This means that all components other than the Edutella libraries work against a KAON RDF model, while Edutella can be used as-is against the Jena interface to the same model.

4.7. Analysis of Retrieved Data

Once the lecturer in our use case has collected a certain amount of material automatically from remote resources or from his own computer,

[12] http://www.prolearn-project.org/
[13] http://jena.sourceforge.net

they are stored as instances in the knowledge base of his ontology. The resources will have to be organized according to their topics. Of course, he will not want to spend much time organizing his documents. Moreover he will have to let the ontology evolve according to the resources he has. For this, we describe in this section two solutions which complement each other: a subjective clustering technique and strategies to manage the evolution of the ontology.

4.7.1. Subjective Clustering

As mentioned earlier, the lecturer needs to organize the learning resources available on his computer or remotely (for example, lectures he retrieved with the focused crawler or through Edutella). He is interested in having his documents grouped according to their topics and similarities. However, standard text clustering techniques cluster only using document/term matrices and thus much of the implicit information contained in the language gets lost. A remedy to that problem is to introduce domain knowledge into the clustering process as so-called *background knowledge*. Ontologies contain such information as, e. g., concept relations, which can then be used by the clustering algorithm to group related documents more efficiently. Clustering mechanisms have been developed that allow to provide *subjective views* onto document collections (Hotho et al., 2001, 2003; Hotho and Stumme, 2002), which are based on an underlying ontology. These techniques can be seen as creating views on the clustered resources, thus using the ontology as a means to specify individual interests.

For instance, one view may concentrate on differences and similarities of the content of learning material, while another view may concentrate on its presentation form, or on the levels of skills and experiences needed. The lecturer can then use the first view to select the material which addresses the topics which are most relevant to his planned course. He might then use the second view in order to see how the material is distributed over different types of material like presentation slides, exercise sheets, or online demonstrations. Moreover, it is possible to get an idea of the topic of the cluster by using a visualization technique based on Formal Concept Analysis which was presented in (Hotho et al., 2003). It displays the distribution of the most relevant terms of the various clusters through the use of a lattice displaying the various combinations of terms occurring in the clusters. This combination of the browsing and the clustering results helps the user to under-

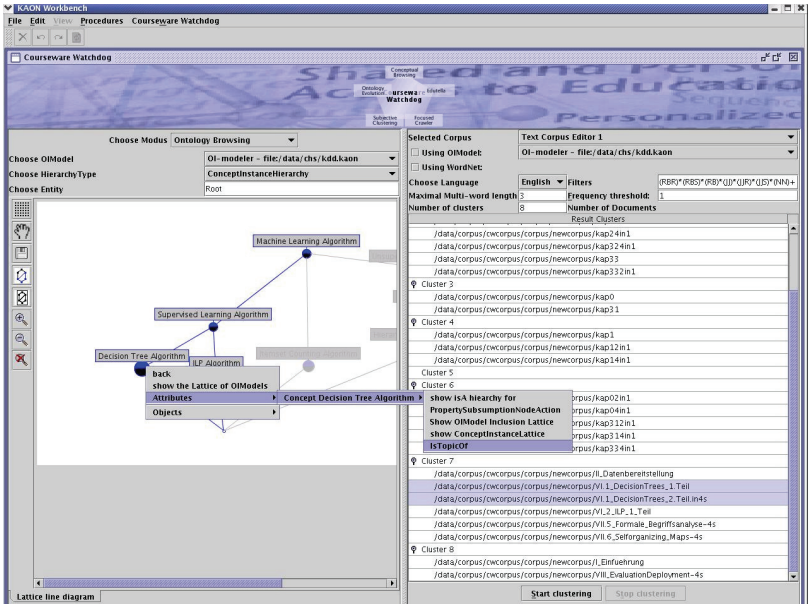

Figure 4.5.: Simple annotation helped by clustering.

stand better the results of the clustering process and select in a simple way the lectures which interest him.

This allows for a very simple interaction with clustering results and achieves the goal of helping the user in organizing his learning material. For example, Figure 4.5 shows how the user, after having selected two documents, can relate them to the concept “Decision Tree algorithm” through the relation “topicOf”.

The subjective clustering is following a five step approach:

1. Parsing the text, collecting interesting term statistics.
2. Building multiword terms (optional).
3. Adding ontological information (various strategies).
4. Clustering documents (with a Bisection K-Means algorithm).
5. Visualization with FCA.

Implementation Details. The first two parts are using the functionalities of TextToOnto (Maedche and Volz, 2001). These are explained in Section 4.7.2. The third part takes the terms found in the documents and tries to

find matching concepts in the ontology. The terms are then weighted using an adapted version of the TF/IDF measure and the new document vector representation can be used in an implementation of the Bisection K-Means algorithm[14]. Finally, the result can be visualized as a lattice displaying the distribution of the clusters according to their most relevant concepts. For this, we reuse the functionalities of the ontology browsing presented earlier in this chapter.

4.7.2. Ontology Evolution

The Courseware Watchdog as presented in this chapter so far builds heavily on a proper ontology that reflects what the user is interested in. However, over time such interests will invariably change together with the teaching/learning subject itself. Therefore, the ontology and the topics represented therein need to be updated. One must deal with several requirements incorporated in such updates:

Modifying the ontology: The ontology must remain consistent at all times. We use the evolution functionalities of the KAON API, which ensure that changes to the ontology will not corrupt it. More details about the maintenance and reuse of evolving and distributed ontologies in KAON can be found in (Maedche et al., 2003; Stojanovic, 2004).

Introducing new concepts: The requirements when introducing new concepts are (a) recognizing that a new concept (e. g. a new topic) has appeared in the course material available in the network or on the Web, (b) inserting this concept into the right place of the taxonomy, and (c) linking it via further relations to other concepts. To that end, we used methods described in (Maedche et al., 2002b) to find relevant concepts.

For instance, Web Services are today an emerging topic, and will probably have to be included in future courses on the Semantic Web. Hence 'Web Services' will be recognized as a term that denotes a new concept, since it occurs frequently in documents on the Semantic Web. It can be inserted into the concept hierarchy (e. g. as a subarea of computer science). It also must be related to other disciplines (e. g. to business process modeling and E-Business). The user can select, on the left hand side, the place where he wants to insert the new concept or instance. Then he can relate the new instances or concepts to other concepts or instances by selecting in the context menu of the new instance the place where he wishes to insert the given concept.

[14]We use the Weka Data Mining framework which we adapted to our needs. For Weka, take a look at: http://www.cs.waikato.ac.nz/ml/weka/.

Implementation Details The technical implementation of the evolution component uses the functionalities of the KAON API to guarantee the coherence of the ontology. In the KAON API, special care has been taken to make sure changes to the ontology reflects changes made by the user while preserving the logical coherence of the ontology level. This has been realized through different means. The user can choose various strategies for the evolution of the model and instance bases. For instance, he can determine if instances of a deleted concept should be deleted also or whether they should be attached to its superconcepts. The users can choose the strategy which is most suitable for his workflow.

The evolution component integrates the use of TextToOnto, a KAON component designed to help users in creating ontologies out of texts (see also Section 4.4.2). The TextToOnto team is still improving the functionalities of the ontology learning functionalities. We are particularly interested how the other Watchdog components can be exploited for ontology learning with TextToOnto.

4.8. Related Work

While the Courseware Watchdog has a more restricted focus, it can be seen as one simple form of Social Semantic Desktop application (cf. Section 2.2.2).

In the e-learning domain, there have been similar efforts aimed at the provisioning of an infrastructure for the exchange of learning objects. Hatala and Richards (2002) describe the Canadian Infrastructure for Learning Repositories, which has a similar structure to the system described here. It is not built around Semantic Web technologies; streamlined metadata schemas are mapped directly to relational databases.

The Conzilla, SCAM, and SHAME systems by Naeve et al. (2005) offer different presentations of RDF-based knowledge repositories which have been applied in the e-learning domain. They integrate with the Courseware Watchdog through the use of Edutella and an application profile of LOM and related standards that has been agreed upon in the PADLR project.

Various learning object repositories have been integrated by using the SQI protocol for querying (Simon et al., 2004). These repositories can be accessed using the Courseware Watchdog as well.

4.9. Conclusion and Outlook

The Courseware Watchdog is a comprehensive approach for supporting the learning needs of individuals in fast changing working environments, and for lecturers who frequently have to prepare new courses about upcoming topics.

As shown, the Courseware Watchdog addresses the different needs of teachers and students to organize their learning material. It integrates, on the one hand, the Semantic Web vision by using ontologies and a peer-to-peer network of semantically annotated learning material. On the other hand, it addresses the important problems of finding and organizing material using semantic information. Finally, it offers an approach to handling the problem of evolving ontologies.

Despite being a prototype, the Courseware Watchdog demonstrates how a Semantic Web based approach increases the support of retrieval and management of remote (learning) resources, by providing tools for discovering and organizing them.

5. Self-Organized Network Topologies for P2PKM

In the previous chapter, we have shown what an end-user application for P2PKM based on Semantic Web technology can look like. In this chapter, we will focus on the problems of query routing *and the* self-organization *of a suitable network topology for P2P networks consisting of such semantic P2P nodes.*
We will base this approach on the idea of building small worlds *of peers in which peers of related contents stay close to each other in the network, thus forming topical clusters. At the same time, these small worlds still retain a low diameter.*
The results in this chapter have been published as (Schmitz, 2004).

5.1. Introduction

The previous chapter has introduced an application with which users can participate in a P2P network in order to exchange learning resources. One particular problem is that of finding one or more appropriate peers which is able to answer a particular user query.

This problem is known as *query routing*. In this chapter, we propose one possible solution for that problem by using the concept of *self organization* as introduced in Section 1.4 in order for the P2P network to assume a structure that is beneficial for routing. In this self organized system of peers, peers reconnect themselves to positions in the network such that the overall fitness of the network for its intended purpose, namely, routing messages efficiently, is optimized.

The resulting structure of the network is called a *small world* (Milgram, 1967; Watts, 1999). This kind of structure has been observed in many real-world networks; these networks exhibit special properties, namely,

a small average diameter and a high degree of clustering, which make them effective and efficient in terms of spreading and finding information.

The main focus of this chapter is to examine the use of small world topologies in an ontology-based P2P knowledge management (P2PKM) system. We assume that in such a system, each peer maintains a knowledge base describing a part of the world relevant to its user. These knowledge bases can then be queried locally or by other peers in the network.

In this chapter, we show a way of letting peers in a P2PKM setting organize themselves into a small world structured around the topics that peers contain knowledge about, and how this small world topology can improve the performance of query routing strategies.

The presented methods are evaluated in a simulation environment. While one possible approach for evaluation would be to instantiate P2P tools such as the Courseware Watchdog in the previous section and observe their behavior, simulation is the standard approach in the P2P community for several reasons. First, it is not feasible to run a non-trivial number of peers, even on larger clusters of computers, due to their aggregated resource consumption (e. g., network sockets, main memory) and the setup overhead. Second, the behavior of such a distributed experiment is intrinsically more difficult to observe than in a simulation. Third, a simulation allows for a more controlled environment than a real-world setting. Thus, most of the published work in the P2P area relies on simulations (see (Aberer et al., 2003a; Crespo and Garcia-Molina, 2002; Stoica et al., 2001) for some examples).

The remainder of this chapter is structured as follows: Sections 5.6 and 5.2 introduce related work and the necessary terms and definitions, respectively. Routing and rewiring algorithms to build a small-world semantic P2P network are described in Section 5.3 and evaluated in Section 5.5. Section 5.7 provides a summary, an interpretation of the results and outlook.

5.2. Basics and Definitions

5.2.1. Model of the P2P network

As this chapter is mainly concerned with network topologies and routing strategies in P2PKM systems, we abstract from the details of a P2PKM system implementation such as elaborated in the previous chapter.

We assume that the following abstractions hold for the system under consideration (similar to (Haase et al., 2004)):

- Each peer stores a set of *content items*, e. g., entities in a knowledge base. On these content items, there exists a *similarity function* sim which can be used to determine the similarity of content items to each other. We assume $d := 1 - sim$ to be a metric in the mathematical sense, i. e., for all content items x, y, z, the following hold: $d(x, x) = 0$, $d(x, y) = d(y, x)$, $d(x, z) \leq d(x, y) + d(y, z)$. The particular set of content items used in this chapter will be entities from an ontology with the related metric such as described in Section 3.4.

- Each peer provides a self-description of what it contains, in the following referred to as *expertise*. Expertises need to be much smaller than the knowledge bases they describe, as they are transmitted over the network and used in other peers' routing tables. In our case, the expertise consists of a content item selected as representative for the peer, but in general, the expertise could also include peer metadata like query languages supported, additional capabilities of the peer etc. As peer expertises are content items, they can be compared to each other and to queries using the sim function.

- There is a relation $knows$ on the set of peers. Each peer knows about a certain set of other peers, i. e., it knows their expertises and network address (IP, JXTA ID). This corresponds to the routing index as proposed by Crespo and Garcia-Molina (2002). In order to account for the limited amount of memory and processing power, the size of the routing index at each peer is limited.

  Sometimes it is more convenient to talk about the network in terms of graph theory. One can view the P2P network as a directed graph $G(V, E)$ with a set V of *nodes* and a set $E \subseteq V \times V$ of *edges*, where each peer P constitutes a node in V, and $(P_1, P_2) \in E$ iff $knows(P_1, P_2)$. We will use both notations synonymously.

- Peers query for content items on other peers by sending query messages to some or all of their neighbors; these queries are forwarded by peers according to some *query routing strategy*. Using the sim function mentioned above, queries can thus be compared to content items and to peers' expertises.

5.2.2. (Weighted) Clustering Coefficients

One observation about small-world networks found in many areas such as sociology or biology is that there are *clusters* of nodes. This means, loosely speaking, that for each node, its neighbors are likely to be connected directly themselves.

More formally, the clustering coefficient for a node v has been defined by Watts (1999) as the fraction of possible edges in the neighborhood of a node which are actually present. We slightly modify that definition to use a directed graph as our *knows* relation may be asymmetric.

$$\gamma_v = \frac{1}{k_v(k_v-1)} \sum_{w \in \Gamma(v)} |\{u \in \Gamma(v) : (w,u) \in E\}| \tag{5.1}$$

where $\Gamma(v)$ are the nodes pointed to by v, not including v:

$$\Gamma(v) = \{u \in V \backslash \{v\} : (u,v) \in E\} \tag{5.2}$$

and $k_v = |\Gamma(v)|$ is the size of the neighborhood. As $k_v(k_v-1)$ in Equation 5.1 is the maximum number of edges possible in the neighborhood, γ_v takes on values between 0 and 1.

The clustering coefficient $\gamma(G)$ of a graph is the mean of the clustering coefficient over all nodes.

In the following, we extend this notion to a *weighted clustering coefficient* γ^w. The motivation for this is that we do not only want to capture how densely connected the neighborhood of each peer is, but also if the neighbors have contents similar to that of the respective peer:

$$\gamma_v^w = \frac{1}{k_v(k_v-1)} \sum_{w \in \Gamma(v)} \mathit{sim}(v,w)\, |\{u \in \Gamma(v) : (w,u) \in E\}| \tag{5.3}$$

This means that for the weighted clustering coefficient of node v, each edge from a neighbor w counts only as much as the similarity between w and v.

The weighted clustering coefficient is related to the observation that in actual small-world networks where there is a notion of similarity between nodes, nodes are not only surrounded by dense neighborhoods. Beyond the density of the neighborhood, the neighbor nodes of a particular node tend to be similar to the node under consideration. In a social network of humans, for example, you are likely to find people of common interests in these clusters. With the above definitions, we have

$0 \leq \gamma_v^w \leq 1$. Large values of γ_v^w mean that v is surrounded by a *dense* neighborhood of *similar* nodes.

Note that other weighted clustering coefficients have been defined (Barrat et al., 2004; Barthelemy et al., 2004; Schank and Wagner, 2005) which do *not* express the same intention as the one defined here.

5.2.3. Characteristic Path Length

The characteristic path length L is a measure for the mean distance between nodes in the network. It is defined by Watts (Watts, 1999; p. 29) as follows: "The *characteristic path length* (L) of a graph is the *median* of the *means* of the *shortest path lengths* connecting each vertex $v \in F(G)$ to all other vertices. That is, calculate $d(v, j)\ \forall j \in V(G)$ and find $\overline{d}_v$ for each v. Then define L as the median of $\{\overline{d}_v\}$." Here, $d(v, j)$ is the number of edges on the shortest path from v to j, and $\overline{d}_v$ is the average of $d(v, j)$ over all $j \in (V - \{v\})$.

For reasons of efficiency, we use the sampling technique proposed by Watts (take a sample $\{v_1, \ldots, v_m\} \subset V$ for some $m < |V|$, compute the mean distance $\overline{d}_{v_i}$ for each, take the median of mean distances as L) to estimate L. Note again that in contrast to (Watts, 1999), we consider our network to be directed.

As the measurement of clustering coefficients and characteristic path lengths requires a global knowledge of the graph, these measures cannot be used directly by the peers to guide their routing rewiring strategies. We will use them instead to evaluate the behavior of the P2P system from the outside.

5.3. Rewiring and Routing Algorithms

In a social system, people tend to be surrounded mostly by people who are similar to themselves in some sense. A librarian will relate to other people who care about books, and a surgeon will probably know some more people from the health care area. In the self-organization terminology from Section 1.4, the social network self-organizes into a state where people tend to be socially connected to people they share some interests with.

This leads to the following observation: if someone wants to find out something about, say, a possible cure for squinting, he may want to ask his friend, the surgeon. Even though he possibly does not know very much about squinting himself, chances are that he will know another

person from the medicine domain who does, e. g., an ophthalmologist. On the other hand, if people are related to very similar people only, this will lead to so-called "caveman worlds" (Watts, 1999), i. e., disconnected cliques which are not connected to each other. In practice, however, many people maintain relationships to people from different professions, geographical locations, etc.; these are called *long range edges*, as they span across multiple communities—densely knit parts of the network that would otherwise be much more distant from each other.

To apply these observations in a peer-to-peer setting and allow our P2P network to organize itself to mimic the behavior of a social system, we need algorithms to make sure that peers can move within the network by establishing new connections and abandoning old ones, trying to get into clusters of similar peers. Peers also need to be able to estimate which of their outgoing edges are most suitable for forwarding an incoming query.

Figures 5.1 and 5.2 show examples of a clustered and an unclustered network, respectively (see Section 5.5.1 for details.); both have been laid out using a spring embedder algorithm. While in Figure 5.1 the nodes are linked randomly, in Figure 5.2, a topical structure can be observed. In the top left, there are peers concerned with persons and research projects (Lecturer, Professor, etc.); in the middle, there are peers containing entities related to organizations; and in the bottom right corner, peers are clustered which deal with publications. In the latter of these two networks, it is reasonable to assume that peers can make "educated guesses" about where to forward any given query, as most peers which will be able to answer can be found in a limited region of the network. In the following paragraphs, we introduce a strategy for clustering P2P networks by topic—namely, rewiring peers greedily to more similar peers they discover on walks through the network. Figure 5.2 is the result of applying the *RandomWalk* strategy to the network of 5.2.

5.3.1. Rewiring Algorithms

In order to build a topically clustered graph using only the kind of knowledge available locally on the peers, we use strategies based on *walks* on the P2P network. A *walk* in this sense is the traversal of a message along a particular path in the network.

To become part of a topically clustered neighborhood, i. e. to be surrounded by similar peers, a peer P_k will periodically initiate the following procedure:

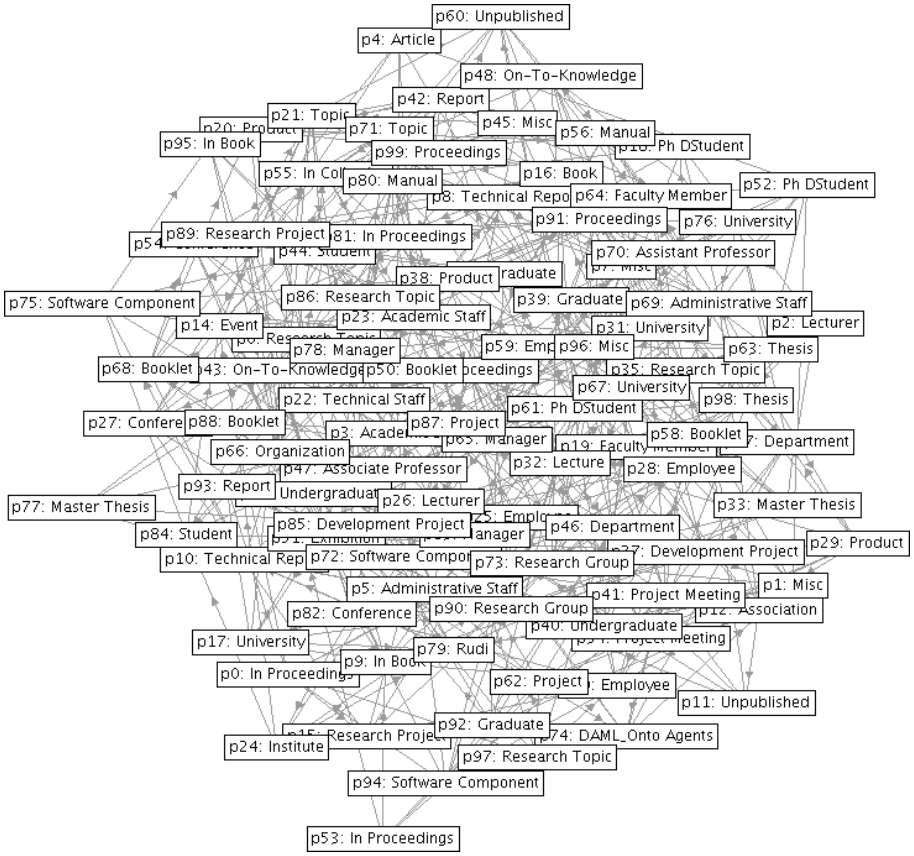

Figure 5.1.: Unclustered network

1. P_k assesses its *knows* relation and decides whether it is in an unsuitable neighborhood, i. e. on the average, its neighbors are too dissimilar from itself:

$$\frac{1}{k_{P_k}} \sum_{P_j \in \{P_j | knows(P_k, P_j)\}} sim(P_k, P_j) < \textit{minSimilarity} \qquad (5.4)$$

   for a given threshold *minSimilarity*.

2. If so, P_k sends a *WalkMessage M*, containing its expertise and a time-to-live (TTL) value, into the network.

3. Message M is forwarded until $TTL = 0$; each forwarding peer appends its own expertise to M and decreases TTL.

4. If $TTL = 0$, M is sent directly back to the original sender P_k.

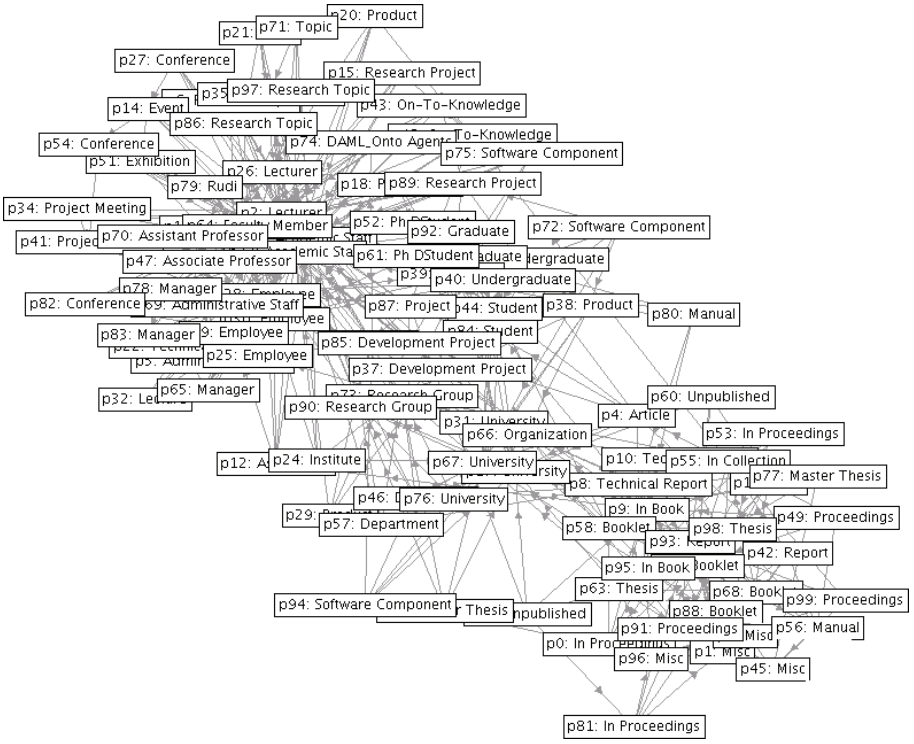

Figure 5.2.: Clustered network

5. P_k collects the other peers' expertises from M. It may find one or more suitable neighbors in that set and decide to keep these in its own routing index, i. e., its *knows* relation. If the routing index size is exceeded, entries for other—less similar— peers may have to be dropped.

The forwarding in Step 3 can be done in different ways:

Random Walk. The message M is forwarded randomly. This is the best one can do if the network is not clustered yet.

Gradient Walk. At each peer P_i, the message M is forwarded to the neighbor of P_i which is most similar to the original sender P_k of M. This is suitable if the network already has a structure corresponding to the ontology (as shown in Figure 5.2); in a random network, however, this strategy will get stuck in local minima too easily.

As with the routing strategies below, these strategies can also be combined, e. g., by choosing the gradient walk and the random walk strategy randomly with equal probabilities (see setting in Section 5.5.1).

5.3.2. Routing Strategies

We have experimented with a number of routing strategies which promise to be useful under the assumptions we made in the preceding sections:

Fixed Fanout Forwarding. The query is forwarded to a fixed number n of neighbors; these are selected to be the n neighbors most similar to the query.

Threshold Forwarding. The query is forwarded to all neighbors which are more similar to the query than a given threshold.

Fireworks. If the query is more similar to the expertise of the forwarding peer than a given threshold, it is broadcast in the neighborhood of the forwarding peer with a new TTL (Ng and Sia, 2002).

Fixed Fanout Random Forwarding. The query is forwarded to a fixed number of randomly selected neighbors.

Random Composite Strategy. A meta-strategy which wraps a number of other strategies plus corresponding weights, and hands over each query to one of the wrapped strategies which has been selected randomly according to the weights. For example, if we wrap strategies A and B with weights 2 and 1, respectively, A will get to handle twice as many queries as B.

Composite Strategy. A meta-strategy using a *chain of responsibility* (Gamma et al., 1995) of strategies; each strategy can claim that it has processed the query, or pass it to the next strategy in the chain. Figure 5.3 shows an example of a composite strategy. First the query is processed locally. Then, the Fireworks strategy gets to handle it, broadcasting it if the necessary level of similarity is met. Otherwise, the query is handled by the Random Composite Strategy, which randomly chooses to hand it over to the Fixed Fanout Strategy.

This way, combinations of different routing strategies can be assembled flexibly. In practice, this will be done by an expert who designs and implements the actual P2PKM system, depending on the requirements; for example, network load can be traded off against recall by increasing the time to live for query messages. One can also imagine a

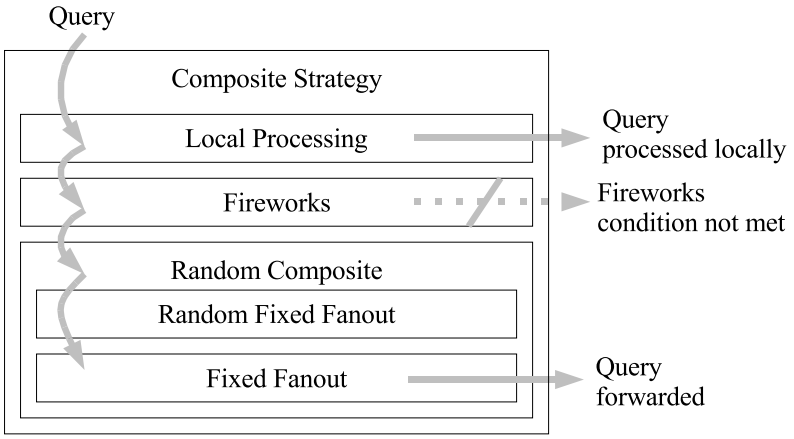

Figure 5.3.: Chained routing strategies

P2PKM system learning appropriate strategies over time (cf. outlook in Section 5.7.2).

Each peer first tries to answer a query from its local knowledge base. Furthermore, each query maintains a list of peers that it has been passed through, so that upon visiting a peer a second time, it will be discarded to avoid cycles.

In principle, the same set of strategies could also be used for the rewiring step in the previous section. For the sake of simplicity, we have restricted the more complex strategies for the routing and used only basic strategies for the rewiring step.

5.4. Implementation Aspects

As real-world P2P networks are hard to instantiate in a non-trivial size and difficult to control, most research in P2P systems is done using simulations.

For the work presented in this chapter, a simulation environment has been implemented that allows for the flexible variation of the components under consideration, e. g., routing and rewiring algorithms, metrics on content items, and many more.

The simulation is based on a discrete event simulation kernel (Frey et al., 2003) written in Java. In that kernel, events such as the sending and receiving of messages are controlled by a central, discrete-time based event queue.

5.5. Evaluation

5.5.1. Setting

As actual P2P networks cannot easily be instantiated with arbitrary numbers of peers, different routing strategies etc., we conducted a number of experiments in a simulation environment. In our experiments, we used 500 peers, each of which was allowed to have a routing index of size 10. Each peer was assigned one randomly chosen entity from the SWRC ontology[1], which acted both as content and as expertise.

The network topology each experiment started with was as a 10-regular random graph, i. e. the *knows* relation of each peer was initialized with the 10 randomly selected other peers. If not stated otherwise, the following parameters were used:

- fireworks strategy with broadcast threshold of 1 (meaning, broadcast only if query matches peer exactly), broadcast TTL of 2,
- if the fireworks strategy does not handle the query, it is forwarded to the two best matching neighbors,
- the query TTL was set to 5,
- the minSimilarity was set to 0.7,
- composite rewiring strategy which randomly chooses between random walk or gradient walk rewiring with equal probability, both with a TTL of 5,

The starts of the rewiring processes at the peers were uniformly distributed over the time interval $[8000, 12000]$; this means that at the beginning of the simulation, only querying activity will take place in the system, but no rewiring. At about 10000 timesteps, the rewiring starts. This is to prevent that all peers start rewiring simultaneously, leading to an unrealistic, synchronous network load. The time between invocations of the rewiring processes was randomly selected from a normal distribution of 500 with a standard deviation of 100.

We have measured the recall—the fraction of retrieved matching entities over the total number of matching entities present in the network—for each query, as well as the total number of messages (query and result messages).

[1] http://ontobroker.semanticweb.org/ontos/swrc.html

5.5.2. Clustering Coefficients

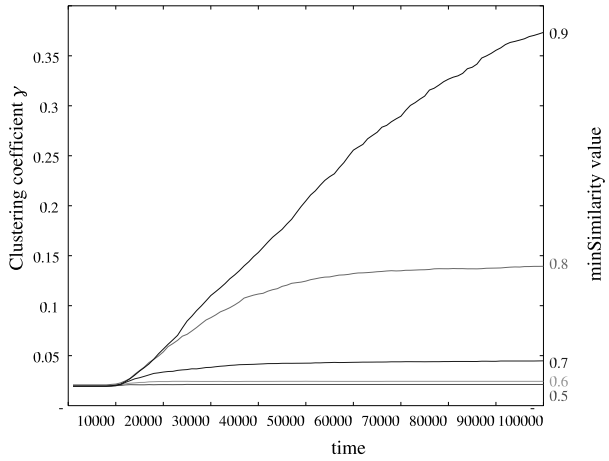

Figure 5.4.: Clustering coefficients over time, for different *minSimilarity* values

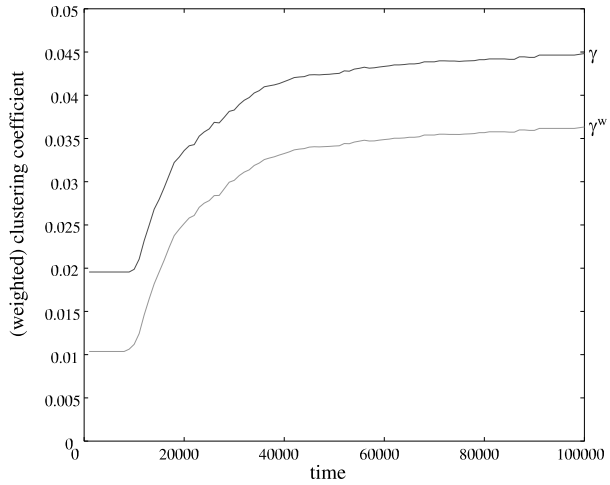

Figure 5.5.: (Weighted) clustering coefficient for *minSimilarity* $= 0.7$

Figures 5.4 and 5.5 show that the clustering coefficients γ increase as intended as a result of the rewiring process. The influence of the *minSimilarity* parameter can be seen: the higher the demands of the rewiring peers are as to the *minSimilarity* of their neighborhoods, the more the clustering coefficients increase. The same holds for the weighted clustering coefficient γ^w, proving that we are building a network of topically related clusters.

Furthermore, note in Figure 5.5 that the weighted clustering coefficient γ^w increases much more than the clustering coefficient (a factor of 3.5 vs. 2.2), relative to the values at the beginning. This is an indication that the clusters that are formed actually consist of topically related peers.

5.5.3. The Influence of Clustering on Recall and Network Load

As can be seen in Figure 5.6, the recall of the queries sent by the peers increases as a result of the rewiring—except for the extreme case with *minSimilarity* $= 0.9$, which will be discussed in Section 5.5.4. In the best case for *minSimilarity* $= 0.7$, an increase of 44% (0.27 vs. 0.39 recall) could be achieved.

At the same time, the number of messages needed per result decreases. Figure 5.7 shows the ratio between the number of messages needed to process each query (query and response messages) and the number of items retrieved. At the end of the rewiring process, about 9 messages are needed to retrieve one result, as opposed to 14 at the beginning (a 36% decrease).

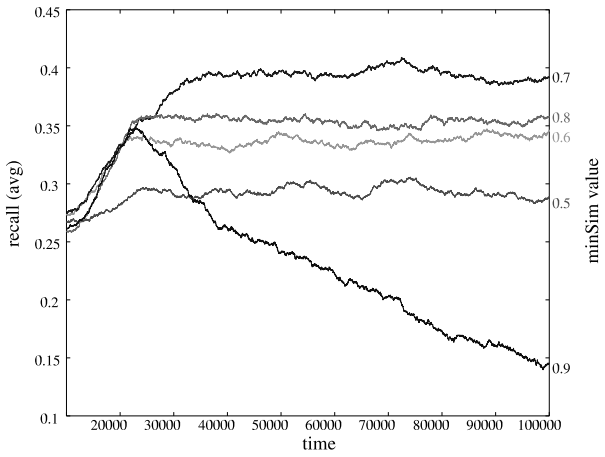

Figure 5.6.: Recall over time, averaged over 10000 timesteps

5.5.4. Clustering Too Much

While the previous section has shown that a certain amount of clustering is beneficial for query routing, it is possible to cluster too much.

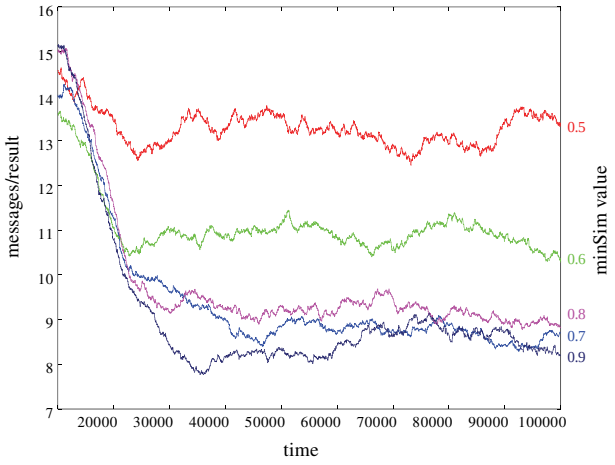

Figure 5.7.: Messages per result obtained, averaged over 10000 timesteps

Figure 5.6 shows the recall for different values of the *minSimilarity* parameter (and thus, different amounts of clustering as shown in Figure 5.4). We can see that for average *minSimilarity* values of around 0.7, the level of clustering achieved is optimal. Lower values do not yield much clustering and improvement in the querying performance at all, while higher values tend to produce clusters which are too tight, thus sacrificing inter-cluster connections. This can lead to "caveman worlds", where each cluster (cavemen in one cave) is very dense, with next to no connections to the outside world. For values of *minSimilarity* close to 1, the graph will even be partitioned into unconnected components.

5.5.5. Characteristic Path Length

As we started from a regular random graph, the characteristic path length of the network was quite small from the outset as is expected from a random graph (Watts, 1999). Although a high clustering coefficient and small L are in many cases contradicting goals (e. g. a hypercube or a random graph have small L, but also small clustering coefficients), Figure 5.8 shows that the characteristic path length was not increased much in the rewiring process, while the clustering coefficient increased (see Section 5.5.2). For the case yielding the best recall—namely, *minSimilarity* $= 0.7$—the characteristic path length increased only about 2%. For a higher *minSimilarity* value, the increase of L was larger. This is a further indication that in that case we cluster too much (cf. Section 5.5.4).

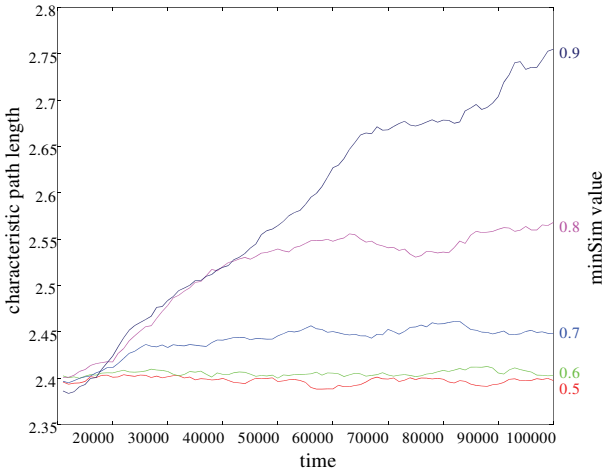

Figure 5.8.: Characteristic path length over time for different *minSimilarity* values

5.6. Related Work

At the time of writing this chapter, there have been several research projects concerned with knowledge management in a P2P setting; examples include Edutella (Nejdl et al., 2002) and SWAP (Ehrig et al., 2003). The assumptions about P2PKM systems made in Section 5.2 are coherent with what has been done in these projects.

A similar view on the rewiring algorithm from Section 5.3 has been presented by Singh and Haahr (2004, 2006) in order to cluster peers by bandwidth or geographical location. They use an analogy to Schelling's model for segregation (1978) to motivate the rewiring step, though the similarity to neighbors is not based on a metric, but on a binary equality criterion—peers either are similar to each other or not.

Much of this chapter draws upon the observations on the structure and evolution of small-world networks presented by Barabási (2003) and Watts and Strogatz (1998). Barabási demonstrates how the structure of existing networks, such as the hyperlink topology of the WWW, can be replicated by a process of *preferential attachment*, meaning that each node in the network will be linked to by others with a probability proportional to its number of links.

Watts and Strogatz (1998) describe the basic notions of the *clustering coefficient* and *characteristic path length* measures as indicators of small-world networks. Watts (1999) examines several models for the growth of small-world networks from what he calls *substrates*: e. g. a substrate

could be a ring or a grid graph from which a small-world topology emerges. He introduces the idea of *rewiring* as a mechanism for evolving the graph structure of a substrate into a more desirable one.

Both Barabási and Watts / Strogatz, however, analyse graphs from a *global view*; for example, to attach preferentially to a node with high in-degree in the Barabási model, a new node would have to see all nodes in the graph in order to assess the respective degrees; on the other hand, a well-connected node is more likely to be found in a random walk. In Section 5.3, we show strategies for rewiring into a small-world network which only rely upon the local knowledge present at each node.

The *knows* relation presented in Section 5.2 is a variant of the *routing index* as presented by Crespo and Garcia-Molina (2002), extended from a keyword-based version into one that contains arbitrary items comparable by a distance function.

Clustering is mentioned as an enabling factor for routing strategies such as *fireworks routing* (Hang and Cheung, 2002) or the superpeer-based routing in Edutella (Nejdl et al., 2003). We complement these works by presenting a strategy for building the necessary clustered structure and demonstrate under which circumstances clustering is beneficial for routing. Hang and Cheung mention a *Learning Fuzzy* method for topology building; this method is not elaborated though, but may be similar to the rewiring strategies mentioned in Section 5.3.

Haase et al. (2004) describe P2P routing strategies based on semantic topologies. In contrast to the work presented here, they use an approach of pushing advertisements of ones own expertise to other peers, as opposed to the rewiring strategies used here which pull information about suitable new neighbors from the network. Furthermore, they do not make any explicit observations about the graph structure of the emergent network.

The idea of self-organizing topical communities has been pursued similarly to this chapter by Tempich et al. (2004). Their method is, however, based on observing past behavior of peers in order to predict their future query answering capabilities, leading, on the downside, to a bootstrapping problem, as peers which have not answered many queries yet will not be found easily.

This idea of extrapolating the past behavior of peers has been extended later. The idea of coupling different routing strategies such that the next one can take over if the previous one fails to find a good routing decision has been re-discovered independently of the work presented here by Löser et al. (2005). What is called “routing strategies” in this chapter is dubbed “network layers” in (Löser et al., 2005), maintaining

the general idea of maintaining fallback strategies if the content based strategy fails.

5.7. Conclusion and Outlook

5.7.1. Conclusion

In this chapter, we have demonstrated how peers in an ontology-based P2P knowledge management scenario can organize themselves into a network topology which reflects the structure of the ontology—i. e., peers having similar contents get to be close to each other in the network, thus forming clusters around common topics. We have provided algorithms which can be executed on each peer without central control to create this kind of topology, and have shown that a clustered topology is beneficial for query routing performance. We have also demonstrated that clustering can be overdone, yielding poorer query results.

Furthermore, we have introduced the notion of a *weighted clustering coefficient* to measure if the clusters that are forming relate to common topics, and provided interpretation of the routing performance with respect to the graph structure of the emerging network.

5.7.2. Outlook and Future Work

The methods described in this paper work well if one assumes that all peers share the same ontology (at least, parts of one ontology large enough so that the shared entities can be used to compute similarities and make routing decisions). This may or may not be realistic, depending on the kind of community which wants to use this kind of semantic P2P network. If the community was not a closely-knit one which can easily agree on some standard ontology for the expected KM tasks, the problems of emergent ontologies, ontology alignment and mapping, standard upper ontologies etc. apply and would have to be solved.

Multiple groups of users, each of which agrees on a standard ontology, would be quite easy to accommodate in a network as described here, however. The use of a certain ontology could be incorporated into each peer's expertise and thus be considered in the routing process.

Our framework for routing and clustering in P2PKM leaves a lot of room for tuning parameters and combinations of strategies. In an implementation of such a network for end-users, one would need to hide all of these parameters and either find default values suitable for a wide

range of possible network states, or find ways for each peer to automatically determine reasonable values, thus enabling the network as a whole to learn suitable parameters.

Furthermore, it is worth discussing which of the parameters should be user-settable. It could be necessary to limit users' possibilities in order to prevent users from accidentally or malevolently flooding the network with query messages and hindering communitcation; on the other hand, a user should be able to express her preferences, e. g. to trade off time against precision.

The clusters in the network can be seen as *communities* of peers with common interests. Making these communities explicit would facilitate tasks such as browsing: if a user finds that peer P_k contains interesting material, she might want to browse the contents of other peers in the community of P_k. Complementary to querying, browsing would provide a different way of accessing the knowledge available in the network. Maintaining and labeling such communities in a decentralized manner would be an interesting extension of P2P knowledge management systems; for example, in the social semantic desktop use case (cf. Section 2.2.2) keeping track of communities is a central aspect.

6. Semantic Summarization of Knowledge Bases for P2PKM

In the previous chapter, we assumed that peers are able to provide a self-description or expertise *of their contents. How this expertise is to be obtained, short of asking the user to provide one manually, is not trivial.*
In this chapter, we develop an algorithm for computing semantic summaries of knowledge bases. These can be used as expertises for peers in a semantic P2PKM setting.
This chapter is based on (Schmitz et al., 2006a).

6.1. Introduction

In Chapters 4 and 5, we have outlined the components of a self-organizing P2P network consisting of peers with knowledge bases modeled in Semantic Web formalisms. The self-organizing and routing methods from Chapter 5 have relied on a self-description or expertise of each peer being available. The two requirements for that self-description are that (a) it must be comparatively small, as it is transmitted over the network and used in other peers' routing indices, and (b) it must represent as well as possible the whole of the contents of that peer.

Thus, we will need a method to *summarize* the contents of knowledge bases, yielding a small set of representative entities that best capture the overall content of the respective knowledge base. Applications of the summarization method presented in this chapter are not restricted to P2PKM applications, however. The same method could be used to summarize knowledge bases in any Semantic Web scenario, e. g., to describe SQI endpoints of learning repositories (cf. Section 4.6.2 and (Simon et al., 2004)).

6.2. Preliminaries and Definitions

6.2.1. Model of a Semantic Peer-to-Peer Network

We assume a P2P network model such as the one described in Chapter 5. Each peer contains a knowledge base modeled using the ontology formalisms described in Section 3.3, including a metric on ontology entities.

6.2.2. Shared and Personal Parts of the Knowledge Bases

We assume that peers do agree on at least a part of an ontology that can be used to describe summaries to each other. More formally, peers $P_i, i = 1 \ldots n$ in the system are assumed to *share* a certain part O of their ontologies: in the case of e-learning, this could be the LOM standard (cf. Chapter 4) plus a classification scheme; when exchanging bibliographic metadata as in Bibster, this could be an ontology reflecting BIBTEX and a classification scheme such as ACM CCS[1], etc.

Additionally, the knowledge base KB_i of each peer P_i contains *personal* knowledge PK_i which is modeled by the user of the peer and is not known a-priori to other peers. Querying this knowledge efficiently and sharing it among peers is the main task of the P2PKM system. Formally, we can say that for all i, $KB_i = O \cup PK_i$.

In Figure 6.1, the ontology used in the evaluation in Section 6.4 is shown. In this case, the shared part O comprises the concepts Person, Paper, Topic, and their relations, as well as the topics of the ACM CCS. The personal knowledge PK_i of each peer contains instantiations of papers and persons and their relationships to each other and the topics for the papers of each individual author in DBLP with papers in the ACM digital library (cf. 6.4.1 for details).

For the purpose of this chapter, the presence of a shared ontology O is assumed. The problem of ontologies emerging in a distributed KM setting (Aberer et al., 2003b), of ontology alignment, mapping, and merging (de Bruijn et al., 2004), are beyond the scope of this work.

6.2.3. k-Modes Clustering

In Section 6.3, we will use an extension of k-modes clustering (Huang, 1998) to obtain aggregations of knowledge bases. The basic version

[1] http://www.acm.org/class

of k-modes clustering for partitioning a set S of items into k clusters $S_1, \ldots, S_k$ such that $S = \dot{\bigcup}_i S_i$ works as follows:

1. Given k, choose k elements $C_i, i = 1 \ldots k$ of S as *centroids*.
2. Assign each $s \in S$ to the cluster S_i with $i = \arg\min_{\mathrm{j}} \mathrm{d}(\mathrm{C_j}, \mathrm{s})$.
3. For $i = 1 \ldots k$, recompute C_i such that $\sum_{s \in S_i} d(C_i, s)$ is minimized.
4. Repeat steps 2 and 3 until centroids converge.

This algorithms yields (locally) optimal centroids which minimize the average distance of each centroid to its cluster members. A variation we will use is *bi-section k-modes clustering*, which produces k clusters by starting from an initial cluster containing all elements, and then recursively splitting the cluster with the largest variance with 2-modes until k clusters have been reached. Compared to the direct splitting into k clusters, bi-section k-modes yields a more uniform distribution of cluster sizes.

6.3. Graph Clustering for Content Aggregation

As mentioned in the motivation, a peer needs to provide an expertise in order to be found as an information provider in a P2PKM network. From the discussion above, the following requirements for an expertise can be derived:

- The expertise should provide an aggregated account of what is contained in the knowledge base of the peer, meaning that using the similarity function, a routing algorithm can make good a-priori guesses of what can or cannot be found in the knowledge base. More specifically, the personal part PK_{i} should be reflected in the expertise.
- The expertise should be much smaller than the knowledge base itself, preferably contain only a few entities, because it will be used in routing indices and in computations needed for routing decisions and will be passed from peer to peer over the network.

With these requirements in mind, we propose the use of a clustering algorithm to obtain an expertise for each peer.

6.3.1. Clustering the Knowledge Base

We use a version of bi-section k-modes clustering for the extraction of such an expertise. As mentioned before, k-modes clustering yields centroids which are locally optimal elements of a set regarding the average distance to their cluster members.

Using a semantic metric as defined in Section 3.4, these centroids fulfill the abovementioned requirements for an expertise: We can compute a *small* number of centroids, which are—on the average—*semantically close* to every member of their respective clusters, thus providing a good *aggregation* of the knowledge base.

In order to apply this algorithm in our scenario, however, some changes need to be made:

- The set S to be clustered consists only of the *personal parts* PK_i of the knowledge bases. Otherwise, the structure of the shared part (which may be comparatively large) will shadow the interesting structures of the personal part.
- The *centroids* C_i will not be chosen from the whole knowledge base, but only from the shared part O of the ontology. Otherwise, other peers could not interpret the expertise of a peer.

The expertise for each knowledge base is obtained by clustering the knowledge base as described, obtaining a set $\{C_i \mid i = 1 \ldots k\} \subseteq O$ of entities from the ontology as centroids for a given k. The expertise then consists of the pairs $\{(C_i, |S_i|) | i = 1 \ldots k\}$ of centroids and cluster sizes.

6.3.2. Determining the number of centroids

One problem of the k-modes algorithm is that one needs to set the value of k beforehand. As the appropriate number of topics for a given knowledge base may not be known a-priori, we use the *silhouette coefficient* (Kaufman and Rousseeuw, 1990), which is an indicator for the quality of the clustering. In short, it determines how well clusters are separated in terms of the distances of each item to the nearest and the second nearest centroid: if each item is close to its own centroid and far away from the others, the silhouette coefficient will be large, indicating a good clustering.

6.4. Experimental Evaluation

In the following sections, we will try to verify three hypotheses:

1. Extracting a good expertise from a knowledge base is harder for large knowledge bases: the interests of a person interested in many areas will be more difficult to summarize than those of someone who has only few fields of interest.

2. With larger expertises, the retrieval results improve: if we spend more space (and processing time) for describing someone's interests, we can make better guesses about what his knowledge base contains.

3. The clustering strategy extracts good expertises with respect to retrieval performance: returning the cluster centroids and counts gives a good approximation of what a knowledge base contains.

6.4.1. Setup

To evaluate the usefulness of the expertise extraction approach from Section 6.3, we consider a P2PKM scenario with a self-organized semantic topology as described in Section 5.2: the expertises of peers are stored in routing tables, where similarity computations between queries and expertises in the routing indices are used to make greedy routing decisions when forwarding queries.

The routing strategies in a P2P network will try to query those peers first that have an expertise close to the query (cf. Section 5.3.2). Thus, a summarization strategy is good if it produces expertises that are similar to a query iff the contents of the knowledge base yield results for that query.

In the following experiment, the quality of the expertises is evaluated in isolation based on that observation: An expertise was extracted for each peer. All of the shared entities of the ontology were used in turn as queries. For each query, the peers were sorted in descending similarity of the closest entity of the expertise to the query. Ties were resolved by ordering in decreasing weight order. We measure the *recall* (defined as in Section 5.5.1) for each query against the *number of peers* which need to be queried for a given recall level.

The evaluation is based on the following use case: there are scientists in the P2P network sharing bibliographic information about their publications. An ontology according to Figure 6.1 is used. Only the top level

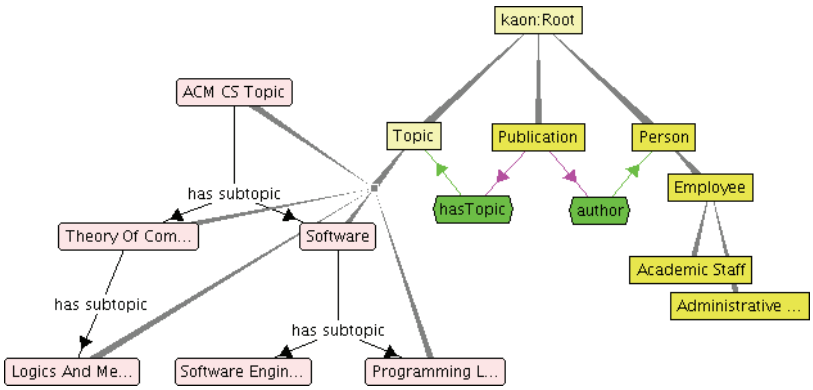

Figure 6.1.: Ontology used in the Evaluation

concepts (Person, Topic, Paper) and the ACM classification hierarchy are shared among the peers. Each user models a knowledge base on his peer representing his own papers.

We instantiated such a set of knowledge bases using the following data and set up the experiment as follows:

- For the 39067 papers from DBLP which are also present in the ACM Digital Library, the topics were obtained from the ACM website. There are 1474 topics in the ACM Computing Classification System. Details on the construction of the dataset and the conversion scripts can be found on http://www.kde.cs.uni-kassel.de/schmitz/acmdata. Each author runs one peer with a knowledge base containing the information about all of his papers.

- To yield non-trivial knowledge bases, only those authors who have written papers on at least 10 topics were considered. This left 317 authors running peers. A discussion of this pruning step can be found in Section 6.4.3.

For each of the summarization strategies described below, we show the number of peers which had to be queried in order to yield a given level of recall. This is an indicator for how well the expertises capture the content of the peers' knowledge bases: the better the expertises, the fewer peers one needs to ask in order to reach a certain level of recall.

This is a variation of the usual precision-over-recall evaluation from information retrieval. Instead of precision, measuring how many of the retrieved documents are relevant, the relative number of the queried peers which are able to provide papers on a given topic is measured.

6.4.2. Expertise Extraction Strategies

In comparison with the clustering technique from Section 6.3, the following strategies were evaluated. The expertise size was fixed to be 5 except where noted otherwise.

Counting (#5): The occurrences of topics in each author's knowledge base were counted. The top 5 topics and counts were used as the author's expertise.

Counting Parents (#P5): As above, but each topic did not count for itself, but for its parent topic.

Random (R5): Use 5 random topics and their counts.

Wavefront (WFL7/WFL9): Compute a wavefront of so-called *fuser concepts* (Hovy and Lin, 1999). A fuser concept is a concept many descendants of which are instantiated in the knowledge base. The intuition is that if many of the descendants of a concept occur, it will be a good summary of that part of the knowledge base. If only few children occur, a better summarization would be found deeper in the taxonomy.

There are two parameters in this computation: a threshold value between 0 and 1 for the *branch ratio*, and a minimal depth for the fuser concepts. There are some problems in comparing this strategy with the others named here:

- It is not possible to control the number of fuser concepts returned with the parameters.
- Leaves can never be fuser concepts, which is a problem in a relatively flat hierarchy such as ACM CCS, where many papers are classified with leaf concepts.
- All choices of parameters yielded very few fuser concepts.

In order to fix these problems, the expertise consists of the fuser concepts as returned by the wavefront computation with the respective number of descendants as weights. If the number of fuser concepts is less than the 5, the expertise is filled up with the leaf concepts occurring most frequently. We examine thresholds of 0.7 (WFL7) and 0.9 (WFL9).

Clustering (C5/C37): The expertise consisted of centroids and cluster sizes determined by a bisection-k-modes clustering as described in Section 6.3. C5 used a fixed k of 5, while C37 selected the best $k \in \{3, \ldots, 7\}$ using the silhouette coefficient.

6.4.3. Results

In this section, results are presented for the different strategies. The values presented are averaged over all queries (i. e. all ACM topics), and, in the cases with randomized algorithms (C5, C37, R5), over 20 runs.

Note that all strategies except C37 returned expertises of size 5, while in C37, the average expertise size was slightly larger at 5.09.

Pruning of the Evaluation Set

In order to yield interesting knowledge bases to extract expertises from, we pruned the ACM / DBLP dataset as described in Section 6.4.1. Thus, only the knowledge bases of authors which have written papers on at least 10 topics were considered.

	Percentage of Peers	
Recall	full data	pruned data
10%	0.01	4.09
30%	0.04	4.93
50%	0.07	6.43
70%	0.16	12.53
90%	0.55	18.73
100%	3.45	22.88

Table 6.1.: Full vs. pruned data: Fraction of peers (%) queried to yield given recall, C5 strategy

Table 6.1 presents a comparison of the full and the pruned dataset for the C5 strategy; the numbers show the percentage of peers that need to be queried for a given level of recall. For example, in the pruned dataset, 12.53% of the peers need to be queried—averaged over all topics—to yield 70% recall. It can be seen that the full data require querying only a fraction of the peers which is one or two orders of magnitude *smaller* than the pruned data, indicating that the first hypothesis holds and the pruning step yields the “hard” instances of the problem.

Influence of the Expertise Size

Intuitively, a larger expertise can contain more information about the knowledge base than a smaller one. In the extreme case, one could use the whole knowledge base as the expertise.

To test this second hypothesis, Figure 6.2 and Table 6.2, show the influence of the expertise size on retrieval performance for the clustering strategy.

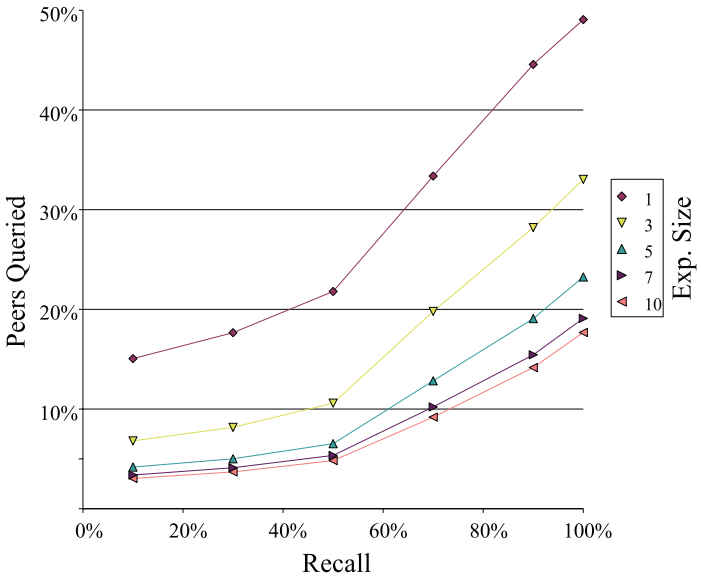

Figure 6.2.: Influence of Expertise Size

	Expertise Size				
Recall	1	3	5	7	10
10%	15.06	6.80	4.09	3.38	3.03
30%	17.66	8.16	4.93	4.12	3.69
50%	21.79	10.59	6.43	5.35	4.82
70%	33.37	19.79	12.53	10.21	9.18
90%	44.57	28.20	18.73	15.44	14.15
100%	49.07	33.04	22.88	19.10	17.67

Table 6.2.: Percentage of Peers Queried against Expertise Size for C5

While the small number of data points for each recall level do not lend themselves to a detailed quantitative analysis, it is clear that the expertise size has the expected influence in the clustering technique: the

Exp. Size	Percentage of Peers
3	20%
4	15%
5	21%
6	23%
7	21%
Avg.: 5.09	

Table 6.3.: Distribution of Expertise Sizes for C37

larger the expertise is, the more detail it can provide about the knowledge base, and the better the retrieval performance is.

Note that the resources a peer would be willing to spend on storing routing tables and making routing decisions are limited, so that a trade-off between resources set aside for routing and the resulting performance must be made.

Influence of the Summarization Strategy

Finally, we evaluate the performance of the clustering strategies against the abovementioned baselines #5, #P5, R5, WFL7, and WFL9.

Table 6.4 and Figure 6.3 show that the k-modes clustering compares favorably against the other strategies: fewer peers need to be asked in order to find a given proportion of the available papers on a certain topic. This is an indication that the clustering technique will yield expertises which can usefully be applied in a P2PKM system with a forwarding query routing strategy based on routing indices. For example, to yield 100% recall, 58% fewer (18.42% vs. 44.15%) peers would have to be queried when using C37 instead of the #5 strategy. With C37 and a routing strategy that contacted best peers first, $100\% - 18.42\% \approx 82\%$ of the peers could be spared from being queried while still getting full recall.

Another noteworthy point is that the parent-counting strategy #P5 performs better than just counting in the #5 strategy. This indicates that even this simple ontology-based aggregation is indeed useful for a knowledge base summary; still, it is far from the performance of C5 and C37.

	Percentage of Peers Queried						
Recall	WFL7	WFL9	C5	C37	#5	#P5	R5
10%	7.96	7.53	4.09	**3.10**	10.69	9.25	6.96
30%	9.55	9.05	4.93	**3.80**	12.15	10.72	8.26
50%	12.30	11.51	6.43	**5.01**	15.33	13.67	11.33
70%	22.86	22.58	12.53	**9.65**	27.43	23.38	24.04
90%	33.86	33.72	18.73	**14.78**	39.45	33.91	35.16
100%	38.89	39.23	22.88	**18.42**	44.15	39.27	39.65

Table 6.4.: Percentage of Peers Queried against Recall

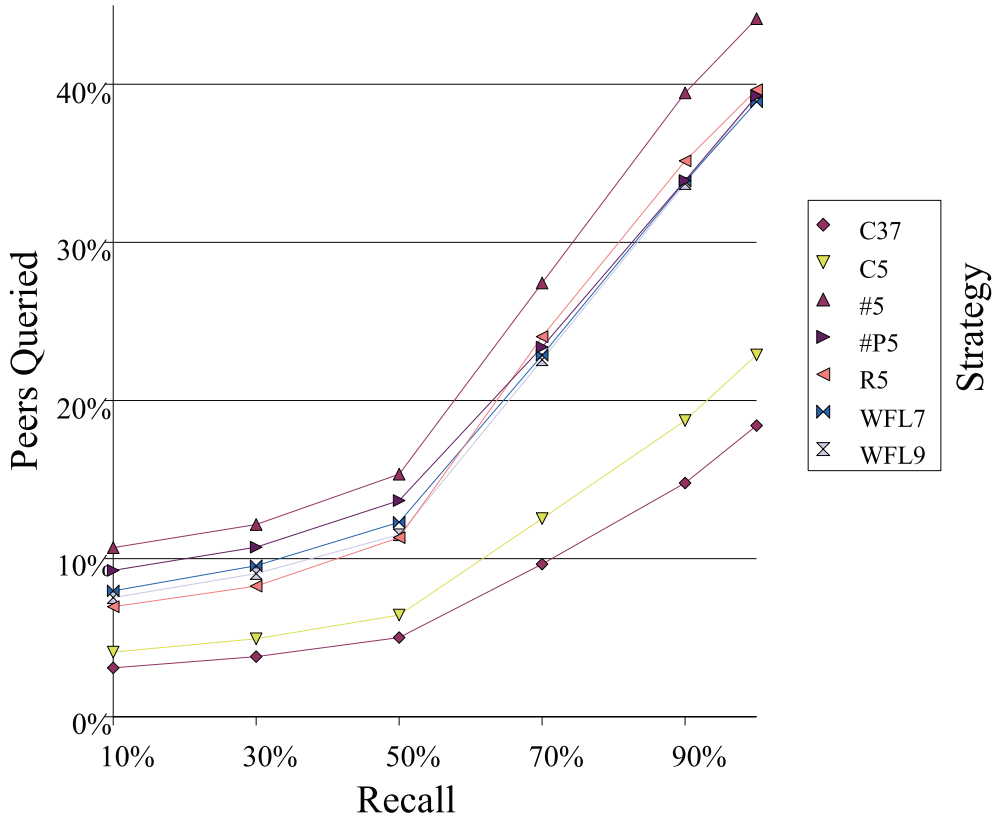

Figure 6.3.: Percentage of Peers Queried against Recall

6.5. Related Work

To the best of the author's knowledge, the particular problem discussed in this paper has not been treated before. There are, however, related areas which touch similar topics.

So-called *knowledge-rich approaches* from the text summarization community (Hovy and Lin, 1999; Hahn and Reimer, 1999) use algorithms on conceptual knowledge representations to extract salient topics from full texts in order to generate summaries. We compare our approach to the one of Hovy and Lin in Section 6.4.

In semantic P2P overlays, peers need some means of obtaining a notion of other peers' contents for routing tables and other purposes. Löser et al. (2005) and others rely on observing the past behavior of peers—queries sent and answered by each peer—to guess what kind of information peers contain, including some fallback strategies to overcome the bootstrapping problem. In (Haase et al., 2004), peers publish their expertise containing *all* topics they contain information about without any aggregation; no strategy is given to limit the amount of routing information each peer publishes.

Keyword-based P2P information retrieval systems make use of the bag-of-words or vector-space models for IR. Reynolds and Vahdat (2003) and others propose the use of Bloom filters to maintain compact representations of contents for routing purposes. These techniques, however, do not provide a semantically aggregated view of the contents, but rather a bitwise superposition of keywords which, for example, loses semantic relationships between related keywords.

Much work has been done on graph clustering (e. g., (Pothen, 1997)) in a variety of areas. Most of these algorithms, though, do not readily yield representatives such as the centroids from the k-modes algorithm used in Section 6.3, and / or may not be naturally adapted to the shared-part / personal-part consideration used in this paper.

6.6. Conclusion and Outlook

6.6.1. Conclusion

In this chapter, an algorithm which can be used to extract semantic summaries—called *expertises*—from knowledge bases is proposed. A motivation for the necessity of this kind of summary is given, namely,

that such summaries are needed for routing tables in semantic P2P networks.

We demonstrate that the clustering method outperforms other strategies in terms of queries needed to get a given recall on a set of knowledge bases from a bibliographic scenario. We also show qualitatively that larger knowledge bases are harder to summarize, and that larger expertises are an advantage in determining which peers to query.

6.6.2. Outlook and Work in Progress

Treatment of Literals. Open research questions include the treatment of literals (e. g. looking for an instance of `PhDStudent` with a last name "Schmitz"). While the simultaneous use of schema- or ontology-based routing indexes and indexes on literal values has been proposed before (Nejdl et al., 2003), to the best of our knowledge there has been no work yet on how the two can be integrated in self-organizing semantic topologies. One possibility would be to treat literal values differently, e. g., in a distributed hash table (DHT) overlay. A similar strategy has been pursued, e. g., by Cai and Frank (2004).

Scalability Issues. Computing the metric as described above is computationally expensive, as it needs to compute all-pairs-shortest-paths, which scales as $O(|V|\cdot|E|\cdot\log|V|)$ (Cormen et al., 2001). For large ontologies having tens or hundreds of thousands of nodes, this is prohibitively expensive. In the current evaluation, the shortest paths needed are computed on the fly using Dijkstra's algorithm (Cormen et al., 2001), but for a real-world P2PKM implementation, some faster solution needs to be found.

On the one hand, the metric needs to be computed only once, so the actual cost of computing it is not relevant for the running system; on the other hand, pre-computing the metric does not mitigate the problem very much, because maintaining the shortest path lengths requires $O(n^2)$ storage, which is intractable for large ontologies.

On possible direction of investigation is to look at the actual usage of the metric in a P2PKM system. If the community structure of the network leads to a locality in the use of the metric, caching and / or dynamic programming strategies for the metric computation may be feasible.

Test Data and Evaluation Methodology. Other than in Information Retrieval, for example, there are neither widespread testing datasets nor stan-

dard evaluation methods available for Semantic Web and especially P2P-KM applications. In order to compare and evaluate future research in these areas, standardized data sets and measures need to be established.

Part II.

Knowledge Management in Folksonomies

7. Introduction to Folksonomies

In this chapter, we will introduce folksonomies, a lightweight mechanism for categorizing resources. We will define folksonomy terminology, outline the major features of folksonomy-based systems, and give examples for different kinds of folksonomy applications. Furthermore, we will discuss the different opinions regarding the relationship between folksonomies and ontologies, and evaluate the strengths and weaknesses of folksonomies as they are used today.

The first part of this thesis was concerned with peer-to-peer knowledge management in the Semantic Web, i. e., the sharing of structured knowledge representations in a distributed system. In this part, we will examine a complementary approach, in which very simple annotations are shared by users in a central repository.

Since about the year 2004, a new view on web-based applications has emerged under the name "Web 2.0" (O'Reilly, 2005); the emphasis in these new applications is on data-driven applications, user participation, simplicity, and information reuse. One common denominator of many of these applications is that lightweight metadata annotations of resources, consisting of free-form keywords often called *tags*, are created by large numbers of untrained users. The annotations are created by the *consumers* of resources, whereas in classical annotation settings the *creators* of the contents or dedicated professionals, such as librarians, provide the metadata. While the Web 2.0 label encompasses many more ideas including wikis, blogs, web services, or syndication of contents, the focus of this part is on the collaborative classification of contents.

A number of names for this kind of classification has been used, including *folksonomies, grassroots classification, social classification, social bookmarking, distributed classification, ethnoclassification, open tagging, faceted hierarchy* (Mathes, 2004; Hammond et al., 2005). While there may be subtle differences in the exact usage of these terms in their different areas of origin, e. g., anthropology, cognitive science, or library science,

we will use them synonymously throughout this thesis, and mainly use the term *folksonomy*, as that seems to be the preferred name at the time of this writing for the kind of system we are describing.

In the remainder of this section, we will give a more detailed definition of folksonomies and the formal model of folksonomies we are using in this thesis. Furthermore, we will discuss the advantages and problems of folksonomies, their history, and typical applications and use cases.

7.1. Terminology

Thomas Vander Wal, to whom the creation of the name *folksonomy* is attributed, has defined it as follows:

> Folksonomy is the result of personal free tagging of information and objects (anything with a URL) for one's own retrieval. The tagging is done in a social environment (usually shared and open to others). Folksonomy is created from the act of tagging by the person consuming the information.
>
> (Vander Wal, 2007; *spelling corrected*)

Similar definitions have been made by others, e. g., by Tonkin and Guy (2006) and Hammond et al. (2005):

> A folksonomy is a type of distributed classification system. It is usually created by a group of individuals, typically the resource users. Users add tags to online items, such as images, videos, bookmarks and text.
>
> (Tonkin and Guy, 2006)

> [...] the new link managers tend to use dynamic categorization systems whereby the user annotates links with whatever terms seem most relevant. Links are generally annotated with 'tags', which are free-form labels assigned by the user and not drawn from any controlled vocabulary.
>
> (Hammond et al., 2005)

In Section 8.1, we will give a formal definition of what is considered a folksonomy in this thesis, namely, that there is a number of *users* who attach arbitrary *tags* to a set of *resources*, thus creating a set of *tag assignments*.

7.1.1. Folksonomies and Ontologies

As Part I of this thesis is concerned with P2P applications making use of Semantic Web technology, including ontologies, and this part is about folksonomies, we will briefly discuss the differences and commonalities of folksonomies on the one hand with ontologies as considered by the Semantic Web community (cf. Section 3.3) on the other hand.

In his widely regarded essay on ontologies as opposed to folksonomies,[1] Shirky (2005) sees a fundamental clash between the ideas of ontologies and folksonomies. In his opinion, the point of ontologies is to determine the "right" set of concepts to describe the world, as well as the "right" concept for each resource to be described. He claims that ontologies were heavily influenced by restrictions that no longer apply for digital contents, e. g., that each book in a library can only be shelved at one particular location at the same time, and that these endeavours of finding one correct description for each resource are bound to fail for a number of reasons. These include different political or social backgrounds of the producers and consumers of annotations, different contexts and purposes of the searching and annotating party, and many more. In the same vein, Gendarmi et al. (2007) call traditional ontology engineering procedures "elitist approaches", as these assume the presence of an omniscient ontology engineer who can anticipate all possible uses of the ontology to be designed.

Mazzocchi (2005) follows a more conciliatory approach. In what he calls "folksologies", he envisions folksonomies in their current form being described using Semantic Web languages such as OWL (cf. Section 3.2). In his vision, tags would start out being unique to users (e. g., user A's tag *apple* would be different—i. e., have a different URI—from user B's tag *apple*). Only after users peruse each others annotations and note commonalities or differences between their usage of tags, a consolidation or differentiation can take place. For example, user A could note that he meant *apple* as "fruit" while user B was thinking about "computer manufacturer". By asserting in an OWL statement that A's apple is different from B's, the user could contribute to a cleaning-up phase that would help the folksology converge from a folksonomy to something that resembles an ontology in the stricter sense. A similar idea of folksonomies converging to richer, more structured knowledge representations is presented by Braun et al. (2007).

[1] See Table 11.14 in Section 11.2 for an indication that Shirky's essay is indeed popular in the Semantic Web community.

An operationalization of this emergence of ontologies from folksonomies is proposed by Mika (2005). He applies mining algorithms and network analysis to the folksonomy structure and infers connections between tags or between tags and resources ("concepts" and "instances" in his terminology) automatically.

As regards the popularity of folksonomy-based KM solutions compared to more formal, RDF- or ontology-based solutions, Halpin (2006) observes that when discussing the semantics of RDF in the W3C, "[in] the contest between social meaning and logic, logic won. While the W3C continued to not address the slippery concept of social meaning, social software took off", and further, regarding folksonomies, "While it is unclear if such a technique can be subsumed by [the] logic-based Semantic Web or [is] a low-cost alternative to the Semantic Web, it seems that 'tagging' is here to stay due to its large deployed user-base".

For the purpose of this thesis, it will be sufficient to consider the terms *ontology* and *folksonomy* as defined in the respective chapters. It will be interesting, however, how these two concepts will interact and influence each other, and if there will be an amalgamated view on both in the future that combines the properties of both.

7.1.2. Classification of Folksonomy Systems

While folksonomy-based systems agree on the basic structure of a folksonomy and are similar in terms of usage, there are still differences between them that allow for a classification of these systems along several dimensions. Marlow et al. (2006) discuss seven dimensions in which folksonomy systems can be classified:

Tagging rights: Who is allowed to tag a resource (the owner, invited users, everybody)? This distinction is also known as "broad versus narrow folksonomies" (Vander Wal, 2005), where narrow folksonomies are those where only the owner a resource is assigning tags, whereas in broad folksonomies every user may tag a resource.

Tagging Support: Are tags recommended when posting, and if so, how are these generated?

Aggregation: Are multiple taggings of the same resource with a given tag counted?

Type of Object: What kind of content (images, web bookmarks, etc.) constitutes the resources in the system?

Source of material: Is the content user-supplied or taken from another source?

Resource connectivity: Are there links between resources, such as between web pages in a social bookmarking system?

Social connectivity: Are there explicit links between users?

The social bookmarking systems we will consider in the following chapters (see also the "social bookmarking" use case in Section 7.2.2) are all of the same kind with respect to these dimensions. They allow tagging by everybody, make tag suggestions, and count multiple taggings of the same resource. Taggable resources are web pages (or, more precisely, anything with a URI[2]), so that there usually are connections between resources in the form of hyperlinks. Also, these systems typically allow users to make explicit connections to others, e. g., by declaring other users as their friends, thus providing social connectivity.

7.2. Folksonomies and their Applications

7.2.1. Typical Features of Folksonomy-Based Applications

Considering the wealth of folksonomy tools present on the web today, one can see a number of features which are common to the vast majority of them. Thus, these can be seen as the defining characteristics of folksonomy tools from the technical standpoint:

Pivot Browsing: As a folksonomy consists of users, tags, and resources, folksonomy tools offer browsing capabilities for each of these dimensions. On the page about a particular user, the resources and tags used by that user will be shown. These tags and resources are hyperlinked to tag and resource pages, which in turn list posts for the respective tags and resources, etc. The same holds also for combinations of dimensions: for example, on a user page showing a particular post with its tags, clicking on one of the tags will lead to a page showing all of the posts by this user with this tag. This kind of linking between all dimensions of the folksonomy enables a quick navigation through its contents, even for untrained users.

[2]Some systems may restrict the set of possible URIs, e. g., by only allowing those with an "http" URI scheme.

Slim Posting Interface: One of the main features of folksonomy tools is the ease of use and the minimization of the overhead required to participate. To that end, a folksonomy site will typically have an interface for entering new posts that requires as little effort as possible. The standard technique is to provide so-called *bookmarklets*, small pieces of code that can be put into the bookmark bar of the web browser. If such a bookmarklet is activated by a mouse click, it will send the URL of the currently visited page to the folksonomy service and initiate posting. The activity required from the user is then reduced to entering tags for the current resource. Though tag recommendations may be frowned upon (Tonkin and Guy, 2006), most folksonomy tools recommend tags that could be used for the given resource, thus reducing the typing overhead even further. After tagging, the user will be redirected back to the original page, so that adding information to the folksonomy entails minimal interruption of the web browsing activity.

Feeds and Connectivity: One last important aspect of folksonomy tools is their emphasis on connectivity. Bearing the concept of *mashups* in mind—i. e., useful applications arising from the combination of services from different web sites (Jhingran, 2006)—a typical folksonomy tools offers several import and export facilities. For example, a folksonomy site will usually offer an RSS feed (RSS Board, 2006) for *every* page it shows. These feeds can be imported by other applications, in the simplest case by stand-alone feed readers. By subscribing to the feed for a tag, one will be presented with all posts for that tag; subscribing to a user-tag combination will show all posts for that user bearing that tag, and so on.

7.2.2. Use Cases and Existing Applications

There is an abundant number of social bookmarking tools available on the web; for example, "All Things Web 2.0"[3] listed 143 bookmarking services at the time of this writing. To give an impression of what typical folksonomy tools look like, we outline three use cases and describe matching systems.

[3] http://www.allthingsweb2.com/mtree/BOOKMARK_2.0/, checked March 15, 2007

Social Bookmarking

Among the very first applications of folksonomies that gained widespread use was social bookmarking. The idea behind these systems is that when browsing the web, a user uploads bookmarks and annotations in the form of free form tags to a centralized web site instead of storing them in his browser. This enables him to access the bookmarks from any computer with internet connectivity; additionally, bookmarks can be shared among users.

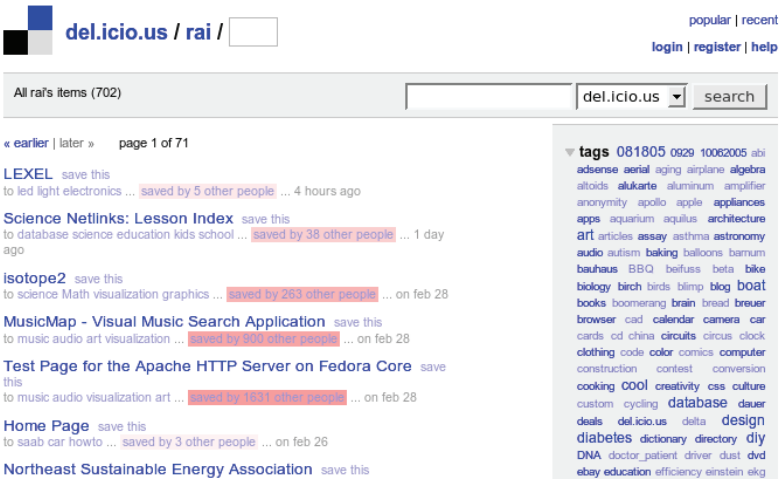

Figure 7.1.: Del.icio.us Screenshot

Figure 7.1 shows a typical page from popular del.icio.us[4] social bookmarking website, established in 2003. On this page, some of the bookmarks of a particular user are shown on the left-hand side. Navigation elements to tag pages and resource pages showing all posts for a particular URL can be seen. On the right-hand side, there is a so-called *tag cloud* showing all tags of that user; the number of occurrences of each tag is indicated by its size and color within the tag cloud.

Social bookmarking is not limited to web resources only. The BibSonomy[5] system—in which the author is involved—is an an example of a social bookmarking system with additional capabilities, in this case the management of references to literature in the BibTeX format. Figure 7.2 shows a screenshot of BibSonomy in which literature references are displayed alongside the web bookmarks.

[4] http://del.icio.us/
[5] http://www.bibsonomy.org/

Even somewhat exotic resource types are managed in folksonomies. In 43 Things[6], for example, users collaboratively manage their personal plans and wishes in life (“learn Spanish”, “lose weight”, or even “get married”); similarly, 43 People[7] allows people to share their wish of meeting celebrities—Johnny Depp, Bill Clinton, Steve Jobs, Conan O’Brien, and Jesus Christ being the favorites,[8] in that order.

Figure 7.2.: BibSonomy Screenshot

Multimedia Resource Management

Another early use of folksonomies in the web was the management of personal media collections. Sites such as Flickr[9] for digital images, Last.fm[10] for audio content, or YouTube[11] for videos are examples of narrow folksonomies.

On these sites, users can upload their multimedia objects in order to present them to the public, and assign tags to them. As opposed to social bookmark management, each resource will be uploaded by one user only (disregarding the possibility of duplicates, which will not be

[6] http://www.43things.com/
[7] http://www.43people.com/
[8] checked April 26, 2007
[9] http://www.flickr.com/
[10] http://www.last.fm/
[11] http://www.youtube.com/

resolved by these services), and only the owner can assign tags to each resource.

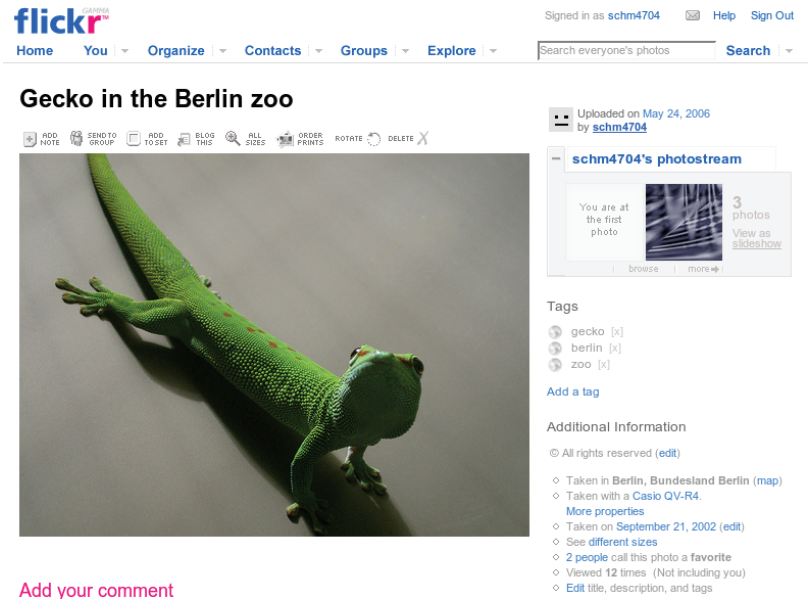

Figure 7.3.: Flickr Screenshot

Figure 7.3 shows a typical page of the Flickr site. A photo is shown together with information about its owner and the tags assigned by the owner. Other users can declare a photo as one of their favourites or leave comments, but assigning tags is not possible for others.

Interestingly, while tools for broad folksonomies such as Del.icio.us typically make it easy for users to copy other users' contents into their own personomy[12], narrow folksonomy tools do not offer this possibility. One can assume that copying is not offered because there is a sense of ownership (and possibly copyright) when uploading multimedia resources, while tagging a web site in a bookmarking service does not mean that the tagging user is the owner of that site in any sense.

Tagging Personal Information: File Systems

As Tonkin (2006) points out in her discussion about the origins of folksonomies, adding lightweight metadata has a history that reaches farther back than the current wave of Web 2.0 applications.

[12]The term *personomy* will be clarified in Section 8.1

One example where keyword-based metadata has been used is in filesystems. A filesystem is the part of an operating system that is responsible for organizing files on secondary storage. Typically, filesystems use hierarchies of so-called *directories* or *folders* to store files. Still, the operating systems and HCI communities have proposed numerous alternative paradigms for organizing, browsing, and searching for files on a personal computer (see (Tonkin, 2006) for references).

As an example, in the Macadam system (Dourish et al., 2000), the idea of *placeless documents* that were retrieved not by their location, but by user-specified properties, was proposed. Interestingly, the creators of the Macadam system already highlighted that "document properties are expressed relative to the *consumer* of the document, rather than the *producer*," an idea that re-surfaces today in the architecture of folksonomies.

On the other hand, while the idea of attribute-based retrieval of files has been proposed (and implemented) years ago, mainstream operating systems do not support similar operations yet; the emphasis is rather on full-text search (e. g., Beagle in Linux, the Spotlight search tool in Mac OS X, or Windows Search in Windows Vista).

Recently, there have been efforts to add tagging capabilities to mainstream operating systems by using lightweight approaches that integrate into the standard filesystem operations, e. g., by Ferré and Ridoux (2001) who navigate files via a formal context generated from tags, or by Bloehdorn et al. (2006) who use WebDAV (Goland et al., 1999) to offer tagging capabilities on files.

7.3. Advantages and Problems of Folksonomy-Based Applications

In order to understand why folksonomy-based applications have gained such a quick acceptance with web users, we will have a closer look at their advantages and disadvantages (cf. (Mathes, 2004; Hammond et al., 2005; Tonkin and Guy, 2006)).

7.3.1. Advantages

Simplicity: One of the most important points in favor of folksonomies over other forms of organization of contents is the simplicity of use. As it takes literally only seconds to upload and annotate a resource in a folksonomy system, and as there are no restrictions

on tagging and a very small learning curve, vast numbers of users are actively participating on a daily basis.

Community Effect and Immediate Feedback: Due to the large number of participating users, posting contents to a folksonomy system yields immediate benefits. Users will be linked to like-minded other users, they can discover related resources matching their tags, and learn what the trends and tendencies within their community are.

Serendipitous Discovery: Thanks to the community effects and the ease of navigation due to pivot browsing, there is a high probability of making "serendipitous discoveries", i. e., finding resources that are relevant to the user's needs although the user did not even now these resources existed and should be searched for.

Desire Lines and Emergent Semantics: While some see a fundamental dichotomy between structured, top-down approaches to organizing contents as opposed to grass-roots efforts such as folksonomies (Shirky, 2005), other argue that folksonomies could be used as a starting point for more structured taxonomies or ontologies (Merholz, 2004; Mazzocchi, 2005).

Merholz compares folksonomies to the so-called *desire lines* in architecture—paths that are paved onto the premises of, say, a college campus only after the users have trodden them onto the grass wherever they thought it most useful. Another view on this phenomenon is that of *emergent semantics*—folksonomies can be seen as a negotiation process in which agents establish the semantics of tags (Staab et al., 2002; Aurnhammer et al., 2006; Cattuto et al., 2007b).

Unanticipated Uses—Mashups: Most folksonomy systems are designed for easy connectivity; e. g., they offer the possibility to subscribe to the posts of certain users or those with certain tags in the form of RSS feeds, and many provide sophisticated programming interfaces.

Thus, they are open for combination with other tools in so-called *mashups*, i. e., situational applications that combine information from different sources (Jhingran, 2006). As an example, it would be straightforward to use a folksonomy service to enhance a web discussion forum with the ability to tag its entries, or to automatically post each discussion item in the forum also to a folksonomy service.

Handling of Non-Textual Content: Whereas traditional means of organizing and searching information often rely on intrinsic features of the pieces of information under consideration—e. g., textual content represented as a bag-of-words model for retrieval—folksonomies can attach user-supplied metadata in the form of tags to any kind of content, including multimedia data such as images or audio streams.

While considerable research has gone into extracting metadata for multimedia content, such as genre classification for music data (Scaringella et al., 2006) or content-based image retrieval (Liu et al., 2007), obtaining metadata via user-supplied tags is implemented easily and works the same across all kinds of content, so that no sophisticated feature extraction has to be performed and different kinds of content can be handled in the same application.

7.3.2. Problems

Unsurprisingly, the problems of folksonomy systems are mostly related to the uncontrolled use of free-form tags by untrained users, resulting in inaccuracies and ambiguities (Mathes, 2004; Hammond et al., 2005; Tonkin and Guy, 2006):

Synonyms and Homonyms: As there is no linguistic processing of tags such as word sense disambiguation, synonyms and homonyms will not be resolved. A folksonomy system will not distinguish between homonyms such as “jaguar” as a cat vs. “jaguar” as an automobile.

The same holds for synonyms: one user may tag his favourite movies with “movies”, another may use “cinema”, a third one “film”. It has been claimed (Shirky, 2005), however, that one person’s synonyms will make a large difference for the next person.[13]

Abstraction Level: When tagging a particular resource, the user has to decide at which level of conceptual detail he or a potential consumer of his annotation will expect to find this resource.

Whether an image of a pet cat, for example, should be tagged “animal”, “cat”, “F. catus siamensis”, or “Fluffy” depends very much on the intended audience of that annotation.

[13] See (Monaco, 2000) for a detailed discussion of the differences between “movies”, “film”, and “cinema”.

A potential remedy would be to use all of these tags at the same time, increasing, on the other hand, the effort of tagging resources consistently.

Inexactness and Variations of Notation: As the annotations in a folksonomy are made by untrained users from different backgrounds with more or less diligence, there are inevitably spelling mistakes, varying use of language (e. g., American vs. British English), and other inconsistencies such as whether the singular or plural forms of words should be used.

While many folksonomy systems offer recommendations upon typing in tags that can mitigate these problems, it is not clear to what extent the system should support the user in his choice of tags (Tonkin and Guy, 2006).

Compound Words: One particular instance of the notational variations mentioned above is the handling of compound words. Different possible ways are used to tag something with, say, "operating systems research". Some users use separate tags, possibly watering down the intended meaning, especially if the system does not maintain the original tag ordering—"operating systems research" may become "operating research systems", which may evoke considerably different associations compared to the originally used compound. Others use underscores or similar symbols to connect words ("operating_systems"), or resort to so-called camel case ("OperatingSystems").

Ranking: As many of the advantages of social bookmarking tools rely on a large-scale user base in order to achieve the statistical mass to compensate for tagging idiosyncrasies, disagreements, and different viewpoints among users, folksonomy tools need to provide a means for presenting a large number of results to users. For example, the del.icio.us dataset we obtained for July 2005 (see Section 9.1 for details) contained about 154,000 resources which are tagged with "css" (presumably meaning *cascading style sheets*).

As of now, the results to user queries are commonly presented in a time-ordered fashion, the latest one being presented first. Obviously, this will not always be what the user is interested in, as he will desire to obtain the "best" resources in some sense, not the latest ones. As in information retrieval, some sort of *ranking* needs

to be put into place in order to allow users to find highest-quality resources first.

Structuring: One of the credos of the folksonomy community is that the set of tags is *flat*, meaning that there is no hierarchy or structure (e.g. hyper-/hyponyms) on the tags (Mathes, 2004).

On the other hand, it is not uncommon for users to have several hundreds or even thousands of tags. For example, in our July 2005 dataset of del.icio.us, there are 70,581 users. Of these, 10,749 have more than 100 tags, 786 have more than 500, and there are 163 users using more than 1,000 tags.

In order to maintain an overview of ones tags, there have to be means of structuring the tag set. Solutions that have been proposed include so-called *bundles* in del.icio.us, which basically serve as baskets collecting tags for presentation, but fulfill no other function. Another possibility is to *allow* users to have a subsumption hierarchy, e. g., to express that every resource tagged with *python* should also be considered to be about *programming*, without *forcing* them to use it. This is what we have implemented in our own system *BibSonomy*.[14]

Personalization and Browsing Support: The window through which a user can peruse the contents of a folksonomy based system—i. e., the user interface visible at any time—is a rather small one. A user will typically look at the page representing the contents for a particular user, a particular tag, or the combination of two or more tags, showing at most about a dozen posts per screen.

In order to focus on contents which are relevant to a given user, it is necessary to allow users to narrow down the amount of information they are presented when browsing the folksonomy, such that the cognitive load of recognizing relevant material decreases.

One very basic approach is offered by the possibility to *subscribe* to particular pieces of content using RSS feeds. Using this technique it is possible to have, for example, all posts by the user *stumme* tagged with the tags *university* and *kassel* in ones feed reader or mail program. More sophisticated approaches would make use of *user profiles* or *communities* to tailor the amount of information presented to a user to his or her needs.

[14] http://www.bibsonomy.org/

Spam: At the time of this writing, there were 4,015 users in *BibSonomy*, 1,765 of which were (manually) classified as spammers. The advantages of posting spam—links to contents which support the commercial interests of the poster, e. g., to merchandise which he wants to sell—are two-fold. Firstly, the contents are possibly found by users of the folksonomy itself. Secondly, as many folksonomy sites have a high visibility in the indexes of search engines, inheriting a good ranking from the folksonomy site is an easy way to promote the web page in search engines.[15]

As the numbers above show, fighting spam is a serious problem in folksonomy-based sites, and in our experience with running BibSonomy, the number of spammers increases superlinearly with the overall size of the user base.

7.3.3. Solutions Discussed in this Thesis

For some of the abovementioned issues, the following chapters will offer possible solutions. In order to have a foundation for the remainder of this part, we will define folksonomies formally in Chapter 8 and present two large-scale data sets in Chapter 9.

Chapter 10 provides a global analysis of the structure of folksonomies using measures adapted from those in the social network analysis and complex systems communities. The algorithms presented in Chapter 11 provide methods for structuring the folksonomy regarding user preferences. Topical clusters can be found using association rule mining (Sec. 11.3) and extended to fuzzy clusters using FolkRank (Sec. 11.2), which can be used to guide the user in exploring the folksonomy. FolkRank can also be used as-is to compute user-specific rankings for users, tags, and resources the respective user is browsing. The rule mining techniques can furthermore be used to extract relationships between tags, thus providing recommendations for structuring a user's tag set.

[15] Many blogs and discussion forums use the "Nofollow" convention (http://en.wikipedia.org/wiki/Nofollow) or the robot exclusion protocol (http://www.robotstxt.org) to render this approach useless, but spammers flood these sites nonetheless.

8. A Formal Model, Data Structures, and Algorithms for Folksonomies

In order to formalize our notion of folksonomies for the next chapters, we will define a mathematical model for folksonomies. Furthermore, this chapter introduces efficient data structures which are able to cope with large-scale datasets and enable efficient implementations for the most common operations needed on folksonomy data. This formal model was first published in (Hotho et al., 2006a).

8.1. Formal Model

A folksonomy consists of users, resources, tags, and assignments of tags to resources made by users. We present here a formal definition of folksonomies, which is compatible with all folksonomy systems discussed in this thesis.

Some of the folksonomy implementations we have discussed, however, will not make use of all the possibilities and degrees of freedom we introduce here. For example, del.icio.us does not use the $\prec$ relation, but rather so-called *bundles*. These bundles are collections of tags used to structure a user's tags, but they cannot be nested in a hierarchy like the $\prec$ relation defined below. As another example, narrow folksonomies have a unique user owning each resource.

Definition 1 (Hotho et al. (2006a))**.** *A* folksonomy *is a tuple* $\mathbb{F} := (U, T, R, Y, \prec)$ *where*

- U*,* T*, and* R *are finite sets, whose elements are called* users*,* tags *and* resources*, resp.,*
- Y *is a ternary relation between them, i. e.,* $Y \subseteq U \times T \times R$*, called tag assignments (TAS for short), and*

- $\prec$ *is a user-specific subtag/supertag-relation, i. e.,* $\prec \subseteq U \times T \times T$*, called* subtag / supertag relation.

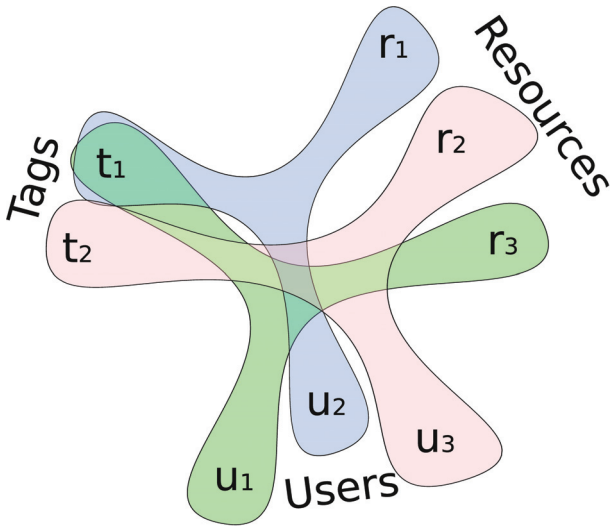

Figure 8.1.: Example Folksonomy with 2 Tags, 3 Resources, 3 Users, and 3 TAS, i. e., Hyperedges

The personomy $\mathbb{P}_u$ *of a given user* $u \in U$ *is the restriction of* $\mathbb{F}$ *to* u*, i. e.,* $\mathbb{P}_u := (T_u, R_u, I_u, \prec_u)$ *with* $I_u := \{(t, r) \in T \times R \mid (u, t, r) \in Y\}$*,* $T_u := \pi_1(I_u)$*,* $R_u := \pi_2(I_u)$*, where* π_i *denotes the projection of an n-ary relation on its* i*th dimension, and* $\prec_u := \{(t_1, t_2) \in T \times T \mid (u, t_1, t_2) \in \prec\}$*.*

A post *consists of all tags assigned by a particular user to a resource. Formally, the set* P *of all* posts *is defined as* $P := \{(u, S, r) \mid u \in U, r \in R, S = T_{u,r}\}$ *where, for all* $u \in U$ *and* $r \in R$*,* $T_{u,r} := \{t \in T \mid (u, t, r) \in Y\}$ *is the set of all tags user* u *has assigned to resource* r*. Thus, a* post *consists of a user, a resource and all tags that this user has assigned to that resource.*

Users are described by a user ID, and tags may be arbitrary strings. What is considered as a resource depends on the type of system. For instance, in del.icio.us, the resources are URLs, and in Flickr, the resources are pictures. From an implementation point of view, resources are represented by some ID, typically an integer such as an MD5 hash of the resource's URL.

If the subtag / supertag relation does not need to be considered, i. e., $\prec = \emptyset$, we will simply note a folksonomy as a quadruple $\mathbb{F} := (U, T, R, Y)$.[1] An equivalent view on folksonomy data is that of a tripartite (undi-

[1] This structure is known in Formal Concept Analysis (Wille, 1982; Ganter and Wille, 1999) as a *triadic context* (Lehmann and Wille, 1995; Stumme, 2005).

rected) hypergraph $G = (V, E)$, where $V = U \dot{\cup} T \dot{\cup} R$ is the set of nodes, and $E = \{\{u, t, r\} \mid (u, t, r) \in Y\}$ is the set of hyperedges.

Figure 8.1 shows an example folksonomy $F = (U, T, R, Y)$ consisting of three users $U = \{u_1, u_2, u_3\}$, two tags $T = \{t_1, t_2\}$, three resources $R = \{r_1, r_2, r_3\}$, and three TAS $Y = \{(u_1, t_1, r_3), (u_2, t_1, r_1), (u_3, t_2, r_2)\}$ visualized as a hypergraph. Note that each of the three hyperedges has exactly one endpoint in the three sets U, T, R.

8.2. Data Structures for Efficient Folksonomy Algorithms

In order to work on large-scale folksonomy datasets and develop high-performance algorithms, an efficient data structure for storing and providing access to folksonomies is necessary.

To meet the needs of the folksonomy algorithms of the following chapters, we devised a data structure based on the ideas of Näher and Zlotowski (2002) for static graphs. Those techniques were generalized to be applicable for the triadic folksonomy graphs as presented in Section 8.1.

8.2.1. Requirements

In the following, we will specify some of the operations that are necessary for our algorithms. We will use the tag set T to discuss the operations; the respective operations for U and R work symmetrically.

TAS for Tag t: Enumerate the set Y_t of tag assignments associated with the given tag t: $Y_t := \{(u, r, t) \in Y \mid u \in U, r \in R\}$

TAS for Pair (t, u): Enumerate the set $Y_{t,u}$ of tag assignments for a given tag t and a given user u: $Y_{t,u} := \{(u, r, t) \in Y \mid r \in R\}$

Count TAS for Tag t: Determine the number $|Y_t|$ of TAS for tag t.

Count TAS for Pair (t, u): Determine the number $|Y_{t,u}|$ of TAS for tag t and user u.

8.2.2. Data Structures and Operations

There are a variety of data structures for graph data types in use today, providing the foundations for different algorithms and accommodating various kinds of real world data.

The folksonomy graphs considered in this part of the thesis are 3-uniform, tripartite hypergraphs (Berge, 1989), i. e., each hyperedge has exactly three endpoints, one in each set U, T, R of nodes. They are large—the largest one we consider has about 3.8 million nodes and 17.4 million hyperedges—and sparse, i. e., the number of hyperedges is much smaller than the maximum number possible: $|Y| \lll |U| \cdot |T| \cdot |R|$. For example, in our del.icio.us dataset (see Section 9.1), we have $|Y| \approx 1.7 \cdot 10^7$, while $|U| \cdot |T| \cdot |R| \approx 1.27 \cdot 10^{17}$, so the density of the folksonomy graph is only about $1.4 \cdot 10^{-10}$. Furthermore, the folksonomies are static as far as our algorithms are concerned.[2] Thus, we are aiming at a representation similar to adjacency lists which is optimised for its memory footprint and adjacency enumeration, at the expense of dynamicity.

Näher and Zlotowski (2002) present an efficient data structure for static graphs, which is used in the LEDA[3] library for efficient data structures and algorithms. It consists of an enumeration of edge endpoints in the adjacency list of each node, in which nodes are listed in ascending order, and an index structure to denote the boundaries of each node's adjacency list in that enumeration. We devised a similar data structure for folksonomies. Other than the LEDA solution from (Näher and Zlotowski, 2002), for folksonomy graphs we need to maintain several permutations of the TAS list Y at the same time, without wasting main memory.

For all computations, the elements of U, T, and R are represented by integers from $\{1 \ldots |U|\}$, $\{1 \ldots |T|\}$, and $\{1 \ldots |R|\}$, respectively. This is achieved by mapping the textual representation of users, tags, and resources, such as depicted in Table 8.1, to a list of TAS in integer form (sometimes called the *fact table*, using OLAP terminology) as in Table 8.2 and dimension tables containing the actual tags, users, and resources. All algorithms presented here and in the following chapters work on the fact table only to save main memory. If, for the presentation of results, a backward mapping of integers to the textual representation is required, it is computed trivially from the dimension tables.

Figure 8.2 gives an example of the data structures that are used to provide efficient implementations of the abovementioned operations. At the core, the elements of Y are represented in an arbitrary order in a $|Y| \times 3$ integer array TAS. The arrays $perm_\sigma$, where σ is a permutation of the dimension names (U, T, R) (in the following identified with the

[2] We are indeed interested in the behavior of folksonomies over time, but for those considerations snapshots at certain intervals are used, each of which is static.

[3] http://www.algorithmic-solutions.com/enleda.htm

Table 8.1.: TAS list in textual form

User	Tag	Resource
schmitz	search	http://www.google.com/
schmitz	university	http://www.uni-kassel.de/
stumme	search	http://search.msn.com/

Table 8.2.: TAS list decomposed into integer TAS list (fact table) and dimension tables

User	Tag	Resource
1	1	1
1	2	2
2	1	3

	User
1	schmitz
2	stumme

	Tag
1	search
2	university

	Resource
1	http://www.google.com/
2	http://www.uni-kassel.de/
3	http://search.msn.com/

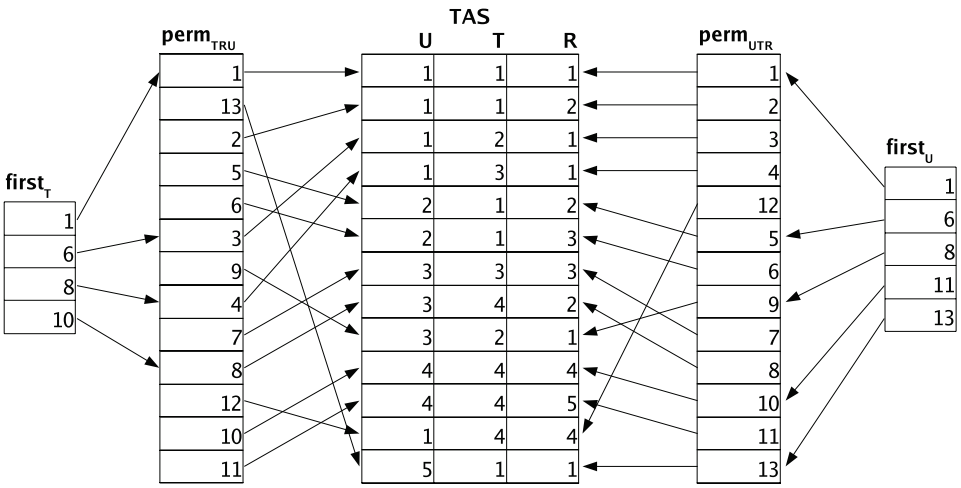

Figure 8.2.: Data Structure for Efficient Folksonomy Operations

integers $(1, 2, 3)$), contain a permutation of the integers $1, \ldots, |Y|$, such that $TAS[\textit{perm}_\sigma[i]]$ is the i-th TAS in lexicographic ordering with respect to the permutation σ of dimensions. Furthermore, the array $\textit{first}_x$ contains in its j-th position the first index i such that $TAS[\textit{perm}_\sigma[i]][\sigma(1)]$ is j, i. e., the position of the first TAS starting with j in the permutation σ of dimensions.

The following time and space costs occur when creating these data structures:

TAS: $3 \cdot |Y|$ integers in main memory, $O(|Y|)$ time for reading from secondary storage.

$\textit{perm}_\sigma$: $O(|Y| \cdot \log |Y|)$ time for reading and sorting, $|Y|$ integers in main memory. If sorted TAS lists are kept on secondary storage, the time costs are reduced to $O(|Y|)$ for reading.

$\textit{first}_X$: $O(|Y|)$ time for collecting the respective first positions, $|X|$ integers of main memory, where X is the respective dimension T, U, or R.

Each additional permutation needs another pair of $\textit{perm}_\sigma$ and $\textit{first}_X$ arrays, at the respective costs. With those data structures in place, the abovementioned operations can be performed efficiently:

TAS for Tag t: Using the $\textit{first}_T$ array of a TRU or TUR ordering, find the first and last positions of TAS for a tag t and iterate over $\textit{perm}_{TUR}$ accordingly. The cost is $O(1)$ for finding the range and $O(|Y_t|)$ for iterating.

TAS for Pair (t, u): Using the $\textit{first}_T$ array of a TUR ordering, find the range where the TAS for tag t are stored. Using binary search, find the first occurrence of user u in that range. Iterate until a TAS with a user other than u occurs. The cost is $O(1)$ for finding the TAS range, $O(\log |Y_t|)$ for finding the first occurrence of u, plus $O(|Y_{t,u}|)$ for enumerating.

Count TAS for Tag t: This operation works the same as *TAS for Tag*, except that the iteration is not needed, lowering the cost to $O(1)$

Count TAS for Pair (t, u): This operation works the same as *TAS for Pair*, at the same cost.

Other operations such as *TAS for user* u or *Count TAS for Pair* (t, r) etc. work symmetrically.

If many computations on the same dataset are to be performed, these data structures could be computed once and stored on secondary storage, although in practice the costs for computing one ordering were reasonably small (about 85 seconds for the largest del.icio.us dataset with $|Y| \approx 17 \cdot 10^6$ on an Athlon 64 machine with 4 GB of main memory, in a straightforward Java 1.5 implementation), so that pre-computing was not deemed necessary.

8.3. Computing Cooccurrence Networks

In Chapter 10, we examine networks of cooccurrence. Tags $t_1, t_2 \in T$ are said to cooccur iff there are a resource r and a user u such that $(u, r, t_1) \in Y \wedge (u, r, t_2) \in Y$. Cooccurence in the other dimensions R and U is defined symmetrically.

In order to compute the cooccurrence networks, we make use of data structures similar to those in Section 8.2. As an example, we consider tag cooccurrence again. In order to compute the (weighted) adjacency list of the tag-tag graph defined by cooccurrence, we first generate a permutation of Y, ordered by URT or RUT, on secondary storage. To compute this permutation, the GNU *sort* utility is used. For very large graphs, we have also used a combination of sorting as large a part of Y as possible in main memory, using a C implementation relying on the standard *qsort* function from the GNU C library, and using GNU *sort* for merging the parts on secondary storage. In that URT or RUT ordering, the tags $t_1, \ldots, t_k$ belonging to a post with k tags can be collected from consecutive tuples in Y. Then, all $\binom{k}{2}$ pairs of tags for that post are enumerated and counted in main memory.

As the size of that enumeration grows as $O(k^2)$, posts with abnormally large k, meaning $k \geq 50$ unless noted otherwise, are discarded. Manual inspection of the folksonomy data shows that posts of that size can be invariably related to spamming activity. Finally, all pairs from the count table which have counts above a user-specified threshold are output to secondary storage, yielding the desired cooccurrence graph.

9. Folksonomy Data Sets

For our experiments, we have used two folksonomy datasets. One has been obtained from the web from the del.icio.us system, while the other one has been taken directly from the database of the BibSonomy system the author is involved in.

In the following chapters, we will make use of two large-scale folksonomy datasets. The first one was obtained from a popular folksonomy site that has been started around 2003 and thus has already acquired a large user base. The second one is from a relatively young system, the development of which the author is involved in. Thus, from that system, a complete dataset could be obtained, whereas the former one is inevitably incomplete to a certain extent, as it is based on screen scraping.

9.1. del.icio.us Dataset

For our experiments, we collected data from the del.ico.us system in the following way. Initially we used `wget` starting from the start page of del.ico.us to obtain nearly 6,900 users and 700 tags as a starting set. Out of this dataset we extracted all users and resources (i. e., del.icio.us' MD5-hashed URLs). From July 27 to 30, 2005, we recursively downloaded user pages to get new resources, and resource pages to get new users. Furthermore we monitored the del.icio.us start page to gather additional users and resources. This way we collected a list of several thousand usernames which we used for accessing the first 10,000 resources each user had tagged. From the collected data we finally took the user files to extract resources, tags, dates, descriptions, extended descriptions, and the corresponding username.

We obtained a folksonomy with $|U| = 75,242$ users, $|T| = 533,191$ tags and $|R| = 3,158,297$ resources, related by in total $|Y| = 17,362,212$ tag

assignments. In addition, we generated monthly dumps from the timestamps associated with posts, so that 14 snapshots in monthly intervals from June 15th, 2004 through July 15th, 2005 are available. To reflect the state of the folksonomy at the given times, the monthly dumps contain all data with timestamps smaller than the respective deadline. Thus, the dump of February 2005 contains the same data as the January 2005 dump, plus the posts of January 16th through February 15th.

9.2. BibSonomy Dataset

As the author is involved in the folksonomy site BibSonomy[1], a second dataset from that system could be obtained directly from a database dump.

As with the del.icio.us dataset, we created monthly dumps from the timestamps, resulting in 20 datasets. The most recent one used in our experiments, from July 31st, 2006, contains data from $|U| = 428$ users, $|T| = 13,108$ tags, $|R| = 47,538$ resources, connected by $|Y| = 161,438$ tag assignments.

[1] http://www.bibsonomy.org

10. Small World Structure in Folksonomies

Social resource sharing systems have acquired a large number of users within the last few years. As a first step to understanding the global structure of these systems from a network analysis point of view, we will analyse the main network characteristics of the two datasets presented in Chapter 9.
We consider folksonomies as tri-partite hypergraphs, and adapt classical network measures to them, namely, the characteristic path length and the clustering coefficient; we then demonstrate that the folksonomies exhibit a small world structure.
The work in this chapter has been published in (Schmitz et al., 2007).

10.1. Introduction

As we have seen in the formal definition of folksonomies in Chapter 8.1, a folksonomy can be viewed as a graph, more specifically: as a tri-partite hypergraph. In this chapter, we will analyze the global structure of folksonomies from a graph-theoretic point of view.

The structure and growth of networks has been a popular research topic recently. With the availability of datasets from massive man-made as well as freely evolving networks, most prominently the World Wide Web, but also collaboration networks, networks of nervous systems, of telephone and power lines, new measures and algorithms for computing these have been developed (Newman et al., 2006). Along the same lines, we will investigate the growing network structure of large-scale folksonomies over time from different viewpoints and with measures

adapted to their hypergraph structure, using two datasets from running systems as examples.

10.2. Small Worlds in Three-Mode-Networks

In Chapter 5, we used the notion of *small worlds* to structure the P2P network; in the following, we will formalize a similar idea on folksonomies. The term *small world* has been coined by Milgram (1967) in the context of social networks, describing the fact that chains of mutual acquaintances between arbitrary persons are often surprisingly short; Milgram, though, does not provide a formal definition for checking whether a network is to be considered a small world. The most common formalization definition of the term is the one by Watts and Strogatz (1998); see also the more comprehensive (Watts, 1999), which we will refer to in the following. They define a *clustering coefficient* and a *characteristic path length*, which describe A *small world network* is then defined as one that has a characteristic the local density of networks around nodes and the typical distance between nodes, respectively. path length comparable to that of a random graph with the same number of nodes and edges, but with a much larger clustering coefficient than such a graph. As the view of folksonomies as graphs is that of tripartite hypergraphs, the common definitions of characteristic path length, clustering coefficient, and random graph do not readily apply; one way to analyze folksonomies is thus to consider projections into ordinary graphs (cf. (Cattuto et al., 2007a) for an example). In the following, we will present corresponding definitions of these concepts on tripartite folksonomy graphs that capture the original intention. We will then check whether the folksonomy graphs from our two datasets exhibit small world behavior and interpret the results in more detail.

10.2.1. Characteristic Path Length

The *characteristic path length* of a graph (Watts, 1999) describes the average length of a shortest path between two random nodes in the graph. If the characteristic path length is small, few hops will be necessary, on average, to get from a particular node in the graph to any other node.

As folksonomies are triadic structures of *(tag, user, resource)* assignments, the user interface of such a folksonomy system will typically allow the user to jump from a given tag to (a) any resource associated with that tag, or to (b) any user who uses that tag, and vice versa for

users and resources. Thus, the effort of getting from one node in the folksonomy to another can be measured by counting the *hyper*edges in shortest paths between the two. Here a path is defined as a sequence of *hyper*edges such that each hyperedge shares at least the user or the resource or the tag with the following hyperedge.[1]

More precisely, let $v_1, v_2 \in T \cup U \cup R$ be two nodes in the folksonomy, and $(t_0, u_0, r_0), \ldots, (t_n, u_n, r_n)$ a minimal sequence of TAS such that, for all k with $0 \leq k < n$, $(t_k = t_{k+1}) \vee (u_k = u_{k+1}) \vee (r_k = r_{k+1})$, and $v_1 \in \{t_0, u_0, r_0\}, v_2 \in \{t_n, u_n, r_n\}$. Then we call $d(v_1, v_2) := n$ the *distance* of v_1 and v_2. We compute path lengths within connected components only. Following Watts (1999), we define $\bar{d}_v$ as the mean of $d(v, u)$ over all $u \in (T \cup U \cup R) - \{v\}$, and call the median of the $\bar{d}_v$ over all $v \in T \cup U \cup R$ the *characteristic path length* L of the folksonomy.

In Section 10.2.3, we will analyse the characteristic path length on our datasets. As computing the characteristic path length is prohibitively expensive for graphs of the size encountered here, we followed the suggestion of Watts (1999; p. 29) and sampled 200 nodes randomly from each graph and computed the path lengths from each of those nodes to all others in the folksonomy using breadth-first search.

10.2.2. Clustering Coefficients

A large amount of clustering or transitivity in a network means that two neighbors of a given node are likely to be directly connected as well, thus indicating that the network is locally dense around each node. To measure the amount of clustering around a given node v, Watts (1999; p. 33) has defined a clustering coefficient γ_v (for normal, non-hypergraphs). The clustering coefficient of a graph is γ_v averaged over all nodes v. He defines the clustering coefficient γ_v as follows ($\Gamma_v = \Gamma(v)$ denotes the neighborhood of v):

> Hence γ_v is simply the net fraction of those possible edges that actually occur in the real Γ_v. In terms of a social-network analogy, γ_v is the degree to which a person's acquaintances are acquainted with each other and so measures the *cliquishness* of v's friendship network. Equivalently, γ_v is the probability that two vertices in $\Gamma(v)$ will be connected.

[1] Paths in hypergraphs are sometimes called *chains* (Berge, 1985), but for symmetry reasons we will use the same terminology as for non-hypergraphs here.

Note that Watts combines two aspects which are *not* equivalent in the case of three-mode folksonomy data. The first one is: how many of the possible edges around a node do actually occur, i. e., does the neighborhood of the given vertex approach a maximally connected graph, i. e., a clique? The second aspect is that of transitivity, i. e., how many pairs of neighbors of a given node are connected by an edge themselves.

Following the two motivations of Watts, we thus define two different clustering coefficients for three-mode data:

Cliquishness: From this point of view, the clustering coefficient of a node is high iff many of the possible edges in its neighborhood are present. More formally: Consider a resource r. Then the following tags T_r and users U_r are connected to r: $T_r = \{t \in T \mid \exists u : (u, r, t) \in Y\}$, $U_r = \{u \in U \mid \exists t : (u, r, t) \in Y\}$. Furthermore, let $tu_r := \{(t, u) \in T \times U \mid (u, r, t) \in Y\}$, i. e., the (tag, user) pairs occurring with r. If the neighborhood of r was maximally cliquish, all of the pairs from $T_r \times U_r$ would occur in tu_r. So we define the clustering coefficient $\gamma_{cl}(r)$ as:

$$\gamma_{cl}(r) = \frac{|tu_r|}{|T_r \times U_r|} = \frac{|tu_r|}{|T_r| \cdot |U_r|} \quad \in [0, 1] \tag{10.1}$$

i. e., the fraction of possible pairs present in the neighborhood. A high $\gamma_{cl}(r)$ would indicate, for example, that many of the users related to a resource r assign overlapping sets of tags to it.

The same definition of γ_{cl} stated here for resources can be made symmetrically for tags and users. For the whole folksonomy, $\gamma_{cl}(\mathbb{F})$ is defined as the arithmetic mean over all elements of $U \cup T \cup R$.

Connectedness (Transitivity): The other point of view follows the notion that the clustering around a node is high iff many nodes in the neighborhood of the node were connected even if that node was not present.

In the case of folksonomies: consider a resource r. Let $\widetilde{tu_r} := \{(t, u) \in tu_r \mid \exists \tilde{r} \neq r : (u, \tilde{r}, t) \in Y\}$, i. e., the (tag, user) pairs from tu_r that also occur with some other resource than r. Then we define:

$$\gamma_{co}(r) := \frac{|\widetilde{tu_r}|}{|tu_r|} \quad \in [0, 1] \tag{10.2}$$

i. e., the fraction of r's neighbor pairs that would remain connected if r were deleted. γ_{co} indicates to what extent the surroundings of the resource r contain "singleton" combinations *(tag, user)* that only occur once. Again, the definition works the same for tags and users, and the clustering coefficient for the whole folksonomy is defined as the arithmetic mean over the nodes.

The following example demonstrates that the clustering coefficients γ_{cl} and γ_{co} do indeed capture different characteristics of the graph and are not intrinsically related. One might suspect that there is a systematic connection between the two, such as $\gamma_{cl}(r) < \gamma_{cl}(s) \Rightarrow \gamma_{co}(r) < \gamma_{co}(s)$ for nodes $r, s \in T \cup U \cup R$, or similarly, on the level of the whole folksonomy, $\gamma_{co}(\mathbb{F}) < \gamma_{co}(\mathbb{G}) \Rightarrow \gamma_{cl}(\mathbb{F}) < \gamma_{cl}(\mathbb{G})$.

However, this is not the case: consider a folksonomy $\mathbb{F}$ with tag assignments $Y_1 = \{(u_1, t_1, r_1), (u_1, t_1, r_2), (u_2, t_1, r_1), (u_2, t_1, r_2), (u_3, t_1, r_3), (u_3, t_2, r_3), (u_4, t_2, r_4)\}$. Here we have $\gamma_{cl}(t_1) \approx 0.556 > \gamma_{cl}(t_2) = 0.5$, but $\gamma_{co}(t_1) = 0.2 < \gamma_{co}(t_2) = 0.5$. Also, there is no monotonic connection when considering the folksonomy as a whole. For the whole folksonomy $\mathbb{F}$, we have $\gamma_{cl}(\mathbb{F}) \approx 0.906, \gamma_{co}(\mathbb{F}) \approx 0.470$. Considering a second folksonomy $\mathbb{G}$ with tag assignments $Y_2 = \{(u_1, t_1, r_1), (u_1, t_1, r_3), (u_1, t_2, r_2), (u_1, t_3, r_2), (u_2, t_1, r_2), (u_2, t_2, r_1), (u_2, t_2, r_2), (u_2, t_2, r_3), (u_2, t_3, r_2), (u_3, t_1, r_2)\}$, we see that $\gamma_{cl}(\mathbb{G}) = 0.642, \gamma_{co}(\mathbb{G}) = 0.669$, thus $\gamma_{cl}(\mathbb{F}) > \gamma_{cl}(\mathbb{G})$ while $\gamma_{co}(\mathbb{F}) < \gamma_{co}(\mathbb{G})$.

10.2.3. Experiments

Setup

In order to check whether our observed folksonomy graphs exhibit small world characteristics, we compared the characteristic path lengths and clustering coefficients to those of random graphs of a size equal in all dimensions U, T, R as well as Y to the respective folksonomy under consideration.

Two kinds of random graphs are used for comparison:

Binomial: These graphs are generated in a fashion similar to an Erdős random graph $G(n, M)$ (Bollobas, 2001). U, T, R are taken from the observed folksonomies. $|Y|$ many hyperedges are then created by picking the three endpoints of each edge from uniform distributions over U, T, and R, resp. This yields a hypergraph in which node degrees are binomially distributed.

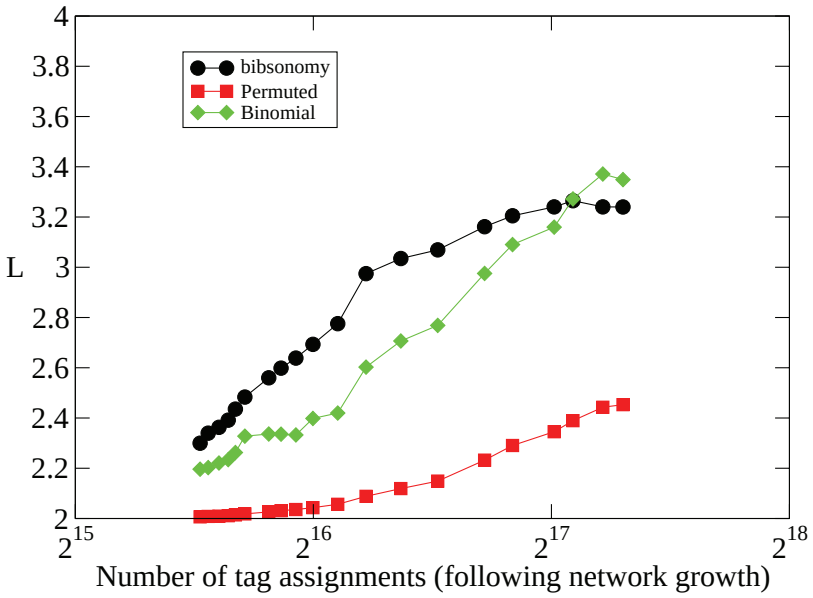

Figure 10.1.: Characteristic path length for the BibSonomy dataset

Permuted: These graphs are created by using U, T, R from the observed folksonomy. The tagging relation Y is created by taking the TAS from the original graph and permuting each dimension of Y independently (using a Knuth Shuffle (Knuth, 1981)), thus creating a random graph with the same degree sequence as the observed folksonomy.

As stated above, the computation of the characteristic path length is prohibitively expensive for graphs of the size encountered here. As for the original del.icio.us and BibSonomy datasets, we sampled 200 nodes randomly from each graph and computed the path lengths from each of those nodes to all other reachable nodes in the folksonomy.

For all experiments involving randomness (i. e., those on the random graphs as well as the sampling for characteristic path lengths), 20 runs were performed to ensure consistency. The presented values are the arithmetic means over the runs; the deviations across the runs were negligible in all experiments.

First Observations

Figures 10.1–10.6 show the results for the clustering coefficients and the characteristic path lengths for both datasets, plotted against the number $|Y|$ of tag assignments for the respective monthly snapshots.

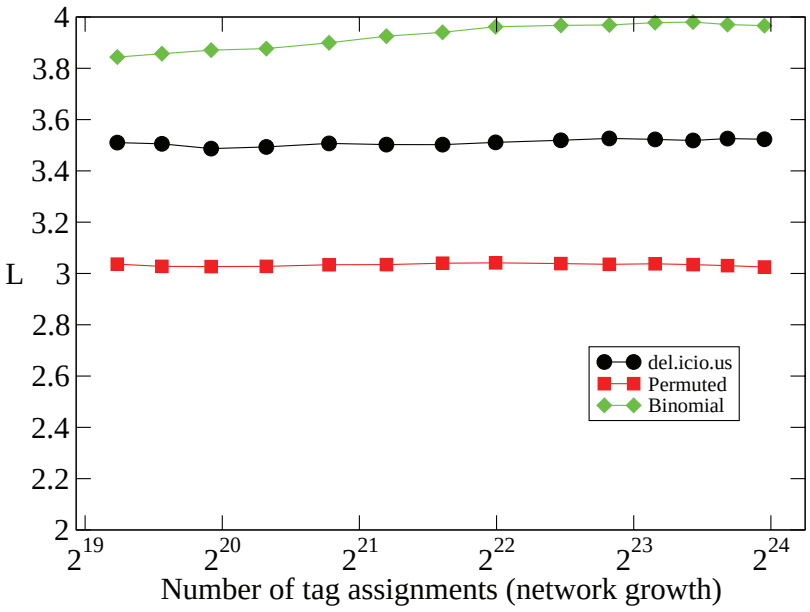

Figure 10.2.: Characteristic path length for the del.icio.us dataset

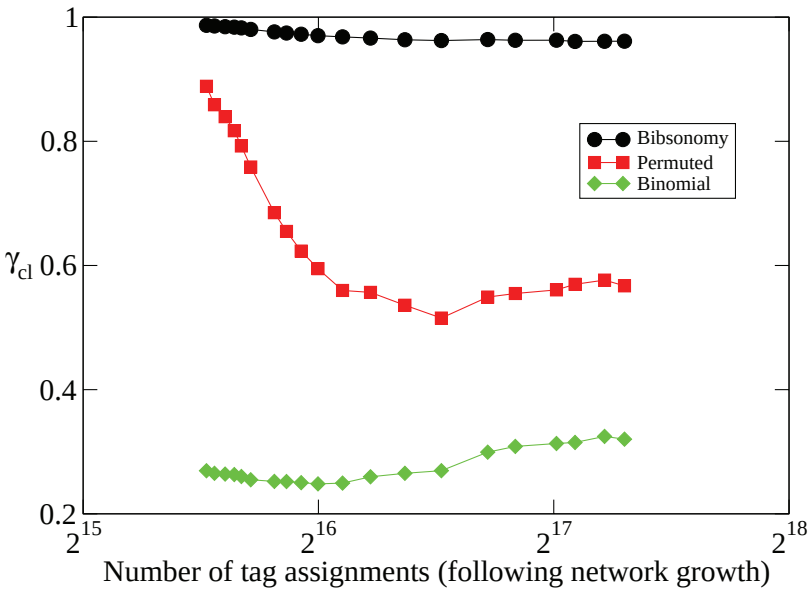

Figure 10.3.: Cliquishness of the BibSonomy folksonomy

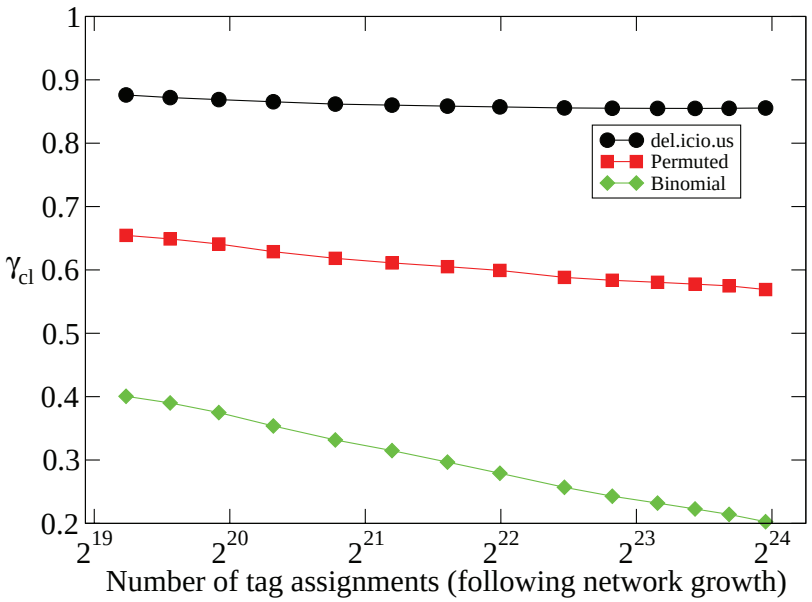

Figure 10.4.: Cliquishness of the del.icio.us folksonomy

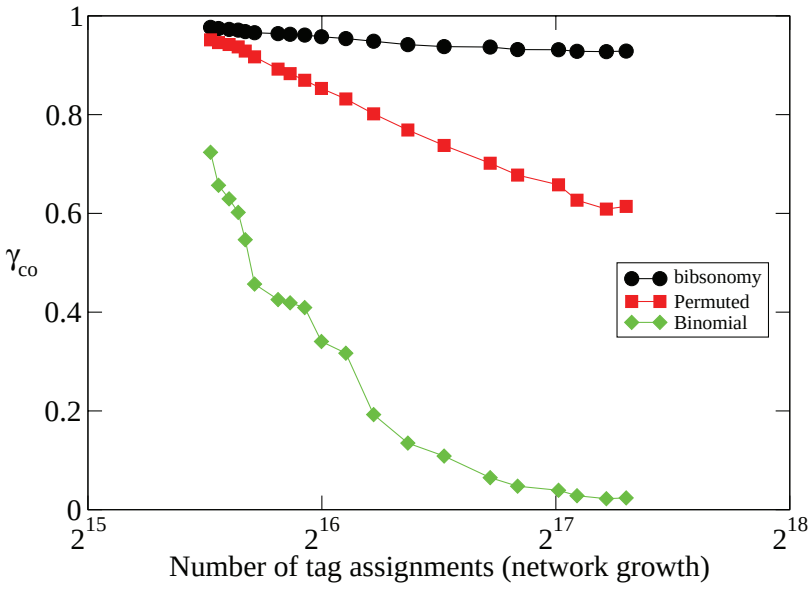

Figure 10.5.: Connectedness of the BibSonomy folksonomy

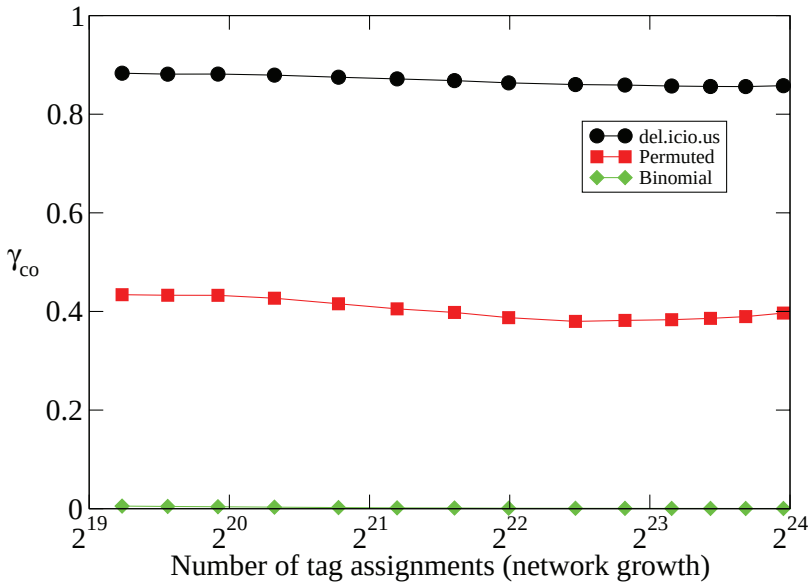

Figure 10.6.: Connectedness of the del.icio.us folksonomy

Both folksonomy datasets under consideration exhibit the small world characteristics as defined at the beginning of this section. Their clustering coefficients are extremely high, while the characteristic path lengths are comparable to (BibSonomy) or even considerably lower (del.icio.us) than those of the binomial random graphs.

Del.icio.us. In the del.icio.us dataset (Figures 10.4 and 10.6), it can be seen that both clustering coefficients are extremely high at about 0.86, much higher than those for the permuted and binomial random graphs. This could be an indication of coherence in the tagging behavior: if, for example, a given set of tags is attached to a certain kind of resources, users do so consistently.

On the other hand, the characteristic path lengths (Figure 10.2) are considerably smaller than for the random binomial graphs, though not as small as for the permuted setting. The comparison with the random binomial graph shows the small world behavior of the human tagging activity. Our interpretation of the comparison with the permuted setting is that the latter maintains the structural features of the human tagging behavior, while introducing additional links between personomies of otherwise unrelated users; leading them thus out of their 'caveman world' (Watts, 1999).

Interestingly, the path length has remained almost constant at about 3.5 while the number of nodes has grown about twentyfold in the observation period. As explained in Section 10.2.1, in practice this means that on average, every user, tag, or resource within del.ici.us can be reached within 3.5 mouse clicks from any given del.icio.us page. This might help to explain why the concept of serendipitous discovery (Mathes, 2004) of contents plays such a large role in the folksonomy community—even if the folksonomy grows to millions of nodes, everything in it is still reachable within few hyperedges, i. e., mouse clicks.

BibSonomy. As the BibSonomy system is rather young, it contains approximately two orders of magnitude fewer tags, users, resources, and TAS than the del.icio.us dataset. On the other hand, the values show the same tendencies as in the del.icio.us experiments. Figures 10.3 and 10.5 show that clustering is extremely high at $\gamma_{cl} \approx 0.96$ and $\gamma_{co} \approx 0.93$—even more so than in the del.icio.us data. At the same time, Figure 10.1 shows that the characteristic path lengths are somewhat larger, but at least comparable to those of the binomial graph. There is considerably more fluctuation in the values measured for BibSonomy due to the fact that the system started only briefly before our observation period. Thus, in that smaller folksonomy, small changes, such as the appearance of a new user with a somewhat different behavior, had more impact on the values measured in our experiments. Furthermore, many BibSonomy users are early adopters of the system, many of which know each other personally, work in the same field of interest, and have previous experience with folksonomy systems. This might also account for the very high amount of clustering.

10.2.4. Characteristic Path Length for Tags

Figures 10.1 and 10.2 demonstrated that the characteristic path lengths L of the two folksonomies under consideration grows comparably to that of the respective "binomial" random folksonomies. As the number of resources $|R|$ dominates the numbers of tags $|T|$ and users $|U|$ by almost one and two orders of magnitude, resp., L is heavily influenced by the characteristic path length for resources.

In order to get an insight into the behavior of tags in that respect, we computed the characteristic path length as described in 10.2.1, but this time taking only the values $\bar{d}_t$ for tags $t \in T$ into account for L.

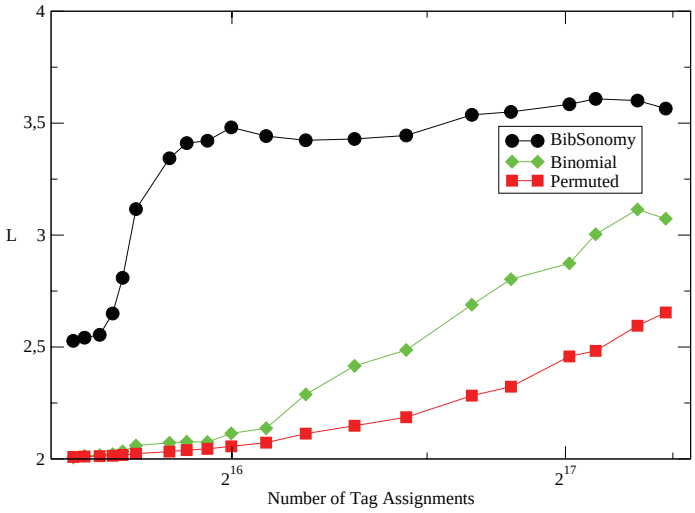

Figure 10.7.: Characteristic path length L considering only tags in BibSonomy

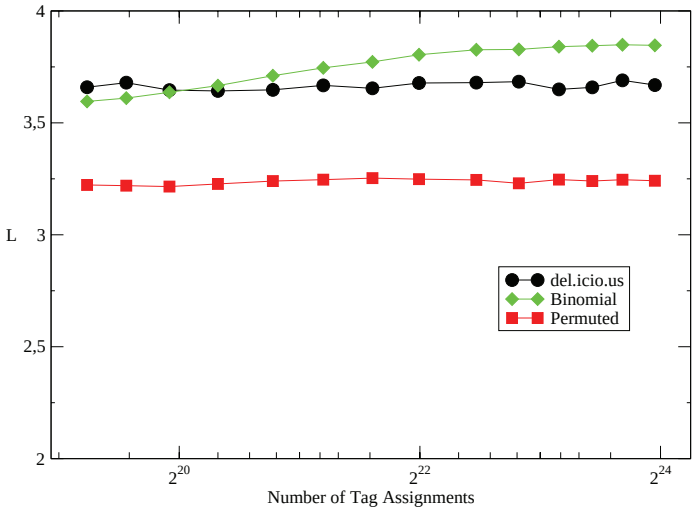

Figure 10.8.: Characteristic path length L considering only tags in del.icio.us

Figures 10.7 and 10.8 show the growth of L for tags in the BibSonomy and del.icio.us folksonomies. Interestingly, the average path length for tags in the BibSonomy dataset is much larger than that for the random folksonomies and rises to about 3.5 to 3.6 very early in the life of BibSonomy, but then remains almost constant. In the del.icio.us folksonomy, which is considerably larger than the latter one, the characteristic path length for tags still remains almost the same at about 3.7.

Our interpretation is that even a small number of early folksonomy users introduces a considerable amount of idiosyncratic vocabulary, of which large parts are rather distant from the rest of the folksonomy. Interestingly, even in the larger del.icio.us folksonomy, the average tag is still farther away from the rest of the folksonomy at $L \approx 3.7$ as opposed to the $L \approx 3.5$ from Figure 10.2 which is largely dominated by resources. This is surprising, as the average tag occurs in about 9 times as many tag assignments as the average resource and would thus be assumed to be better connected to the rest of the network than the average resource.

10.2.5. A Closer Look on del.icio.us

We will conclude this section by a closer look on how the characteristic path length, the cliquishness, and the connectedness are distributed over the users, tags, and resources in del.icio.us. To this end, we have computed the cooccurence graphs for the three dimensions users, tags, and resources as described in Section 8.3. The characteristic path length and clustering coefficients of the (non-hyper) cooccurrence graphs are shown in Figures 10.9 and 10.11. The characteristic path length was approximated by taking a 200-node sample, and for the clustering coefficient the approximation of Schank and Wagner (2005) was used with a precision of $\epsilon = 10^{-3}$ and a probability of 0.99.

Figure 10.9 shows the characteristic path lengths of the respective cooccurence graphs for tags, users, and resources.[2] The result is as expected: the set of resources is almost an order of magnitude larger than the set of tags, which is about the same ratio larger than the set of users. The larger graphs have higher characteristic path lengths.

Figure 10.10 shows the different contributions of the tags, users, and resources to the del.icio.us curves of Figures 10.2 and 10.8. For computing the values, the random nodes have been drawn only from the respective dimensions. The low path length for the user nodes indicates that personomies are a structural element in a folksonomy: Consider

[2] Note that the three values are measured in three different graphs.

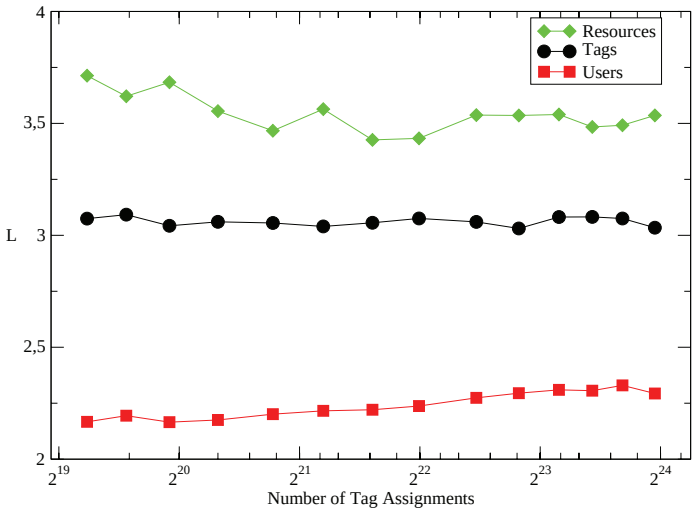

Figure 10.9.: Characteristic path lengths in the three cooccurrence graphs for del.icio.us

the extreme case that all personomies are completely disjoint. Then the users are the central nodes in their connected component (which equals their personomy), and have thus shorter characteristic path lengths in average.

The characteristic path lengths of the tags and resources in the cooccurrence graphs (Figure 10.9) diagram are reversed compared to the hypergraph (Figure 10.10). This is likely to be due to the fact that users tend to invent new, personal tags—which are further away from the core of the folksonomy—whereas there is less divergence of the URLs to be included in the system.

Figure 10.11 shows the clustering coefficient of del.icio.us for the three cooccurrence graphs; Figure 10.12 depicts the connectedness of the hypergraph by dimension. Both diagrams show that the neighborhoods around tags and resources are denser than around users. This is likely to stem from the fact that users usually have different interests. An interesting observation is that the user curve decreases over time in the former diagram, while it increases in the latter. Both effects result from the increasing number of neighbors over time. The clustering coefficient decreases because less and less neighbors are connected to each other when the neighborhood increases. γ_{co} on the hand increases over time,

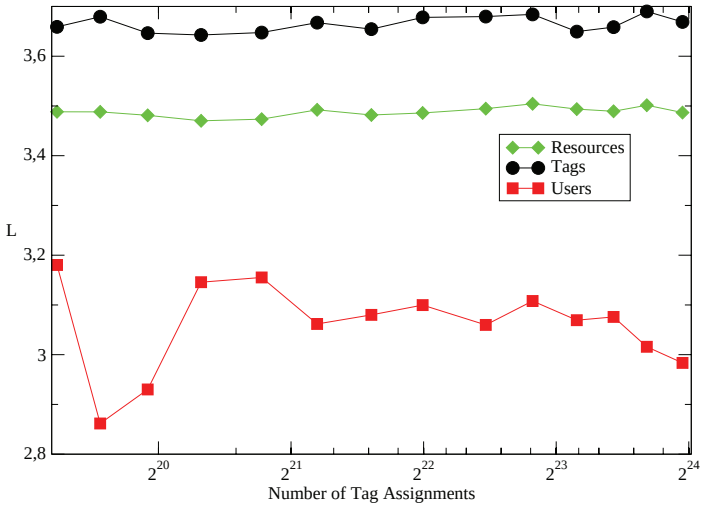

Figure 10.10.: Characteristic path lengths for tags, users, resources in the del.icio.us hypergraph

as it becomes — with increasing neighborhoods — more likely that, for a given TAS, another user has assigned exactly the same tag to the same resource. This indicates that vocabularies of different users converge, resulting in emergent semantics.

Figure 10.13 shows that the cliquishness for tags and resources is high, indicating that if, e.g., a resource is tagged with certain tags by certain users, many of the possible combinations of those tags and users are likely to occur, i.e., there is a natural set of tags which seem appropriate for a given resource, and vice versa, for a given tag, the users using that tag agree to a large extent on which resources should be tagged with it. On the other hand, the cliquishness for users is considerably lower. This demonstrates that, other than tags and resources, users typically have several fields of interest and thus are connected to elements of the other dimensions which will not necessarily occur in many of the possible combinations.

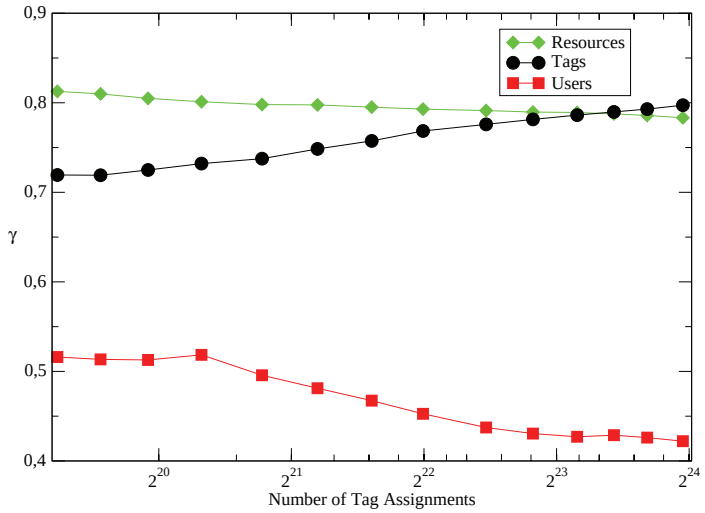

Figure 10.11.: Clustering coefficient of del.icio.us for the three cooccurrence graphs

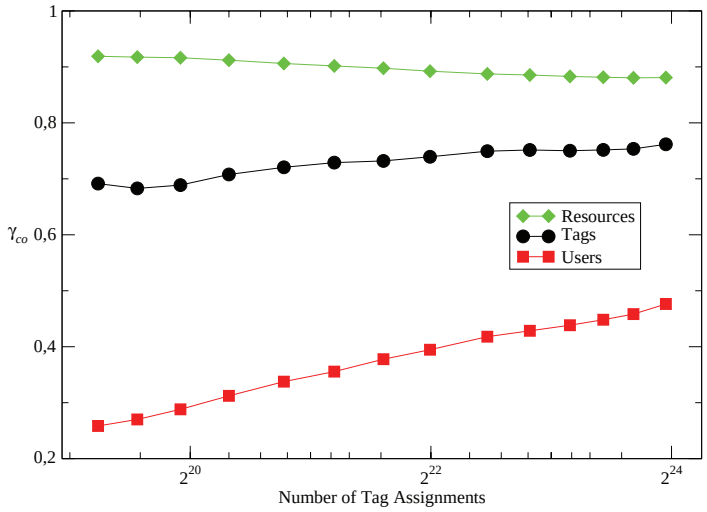

Figure 10.12.: Connectedness γ_{co} for the del.icio.us hypergraph by dimension

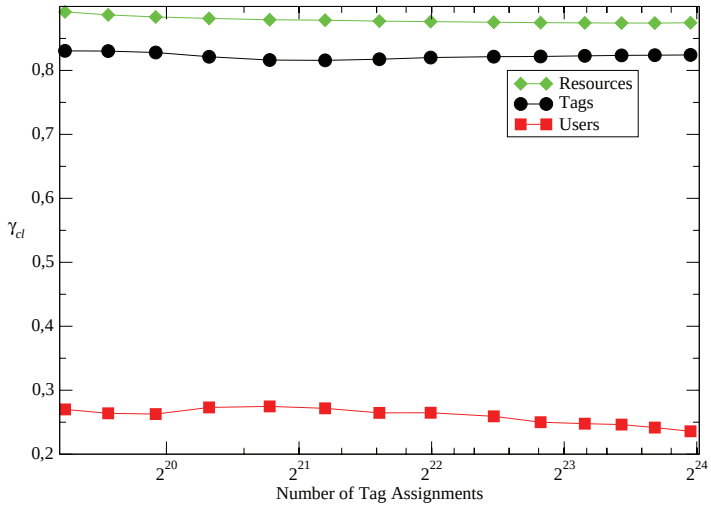

Figure 10.13.: Cliquishness of del.icio.us for the three dimensions in the hypergraph.

10.3. Related Work

10.3.1. Folksonomy Mining

With the recent interest in folksonomies, there have been several contributions that present mining and network analysis approaches on folksonomies, though, to the best of the author's knowledge, a generalization of the small world structure to triadic hypergraphs as presented here has not been done before.

Regarding the structural properties of folksonomies, often projections or aggregations are considered, most often on the tag-tag cooccurrence network (Shepard et al., 2006; Cattuto et al., 2007a; Schmitz, 2006; Cattuto et al., 2007b).

Other mining tasks on folksonomies that have been examined include the discovery of trends in the tagging behavior of users (Dubinko et al., 2006; Hotho et al., 2006b), or learning taxonomic relations from tags (Mika, 2005; Heymann and Garcia-Molina, 2006; Jäschke et al., 2006).

10.3.2. The New Science of Networks

The notions of small worlds, clustering coefficients, and characteristic path lengths, are part of a framework for network analysis that has been called “the new science of networks”. For a comprehensive overview of the most relevant results in that area, refer to (Newman et al., 2006).

In this new line of research, not only the static structural properties of complex networks, but more importantly the dynamic growth processes of complex networks are examined. The results from this chapter are combined with additional measures and observations from complex networks research in (Schmitz et al., 2007).

Variations of the clustering and characteristic path length measures used in this chapter have been presented for special kinds of networks such as bipartite or weighted networks (Lind et al., 2005; Barrat et al., 2004), though no versions applicable for tripartite hypergraphs have been available to date.

10.4. Summary and Outlook

10.4.1. Conclusion

In this paper, we have analyzed the network structure of the folksonomies of two social resource sharing systems, del.icio.us and BibSonomy. We observed that the tripartite hypergraphs of their folksonomies are highly connected and that the characteristic path lengths are relatively low, facilitating thus the “serendipitous discovery” of interesting contents and users. We have further presented some peculiarities that can be observed when drilling down into greater levels of detail, such as the behavior of the measures for the different dimensions and on cooccurrence graphs.

10.4.2. Future Work

Analysis of Transaction Data: So far, this chapter has analyzed the structure of folksonomy graphs as static snapshots taken at certain points in time. For the BibSonomy data, however, we have the full transaction log available which documents each step of users when building up the folksonomy. Thus, one can analyze in more detail how the structure of the folksonomy graph develops; examples include effects of copying of

posts, or of preferential attachment effects that result from more popular resources being visible on more pages of the folksonomy system.

Identification of Communities: As the results from this chapter suggest that the folksonomy consists of densely-connected communities, a second line of research that we are currently pursuing and that will benefit from the observations in this paper is the detection of communities. This can be used, for example, to make those communities explicit which already exist intrinsically in a folksonomy, e. g. to provide user recommendations and support new users in browsing and exploring the system.

11. Information Retrieval and Structure Mining in Folksonomies

This chapter presents ways of exploiting the inherent structure of a folksonomy in order to support the user of a folksonomy system in querying and browsing. The ranking of query results according to user preferences in a folksonomy system is discussed and our ranking algorithm FolkRank *is presented. Exploiting common regularities in the folksonomy between certain items can be used to extract knowledge about local structures, e. g., topical clusters of resources, or subsumption relationships between tags. We make use of rule mining algorithms on the folksonomy to extract such structures. Both techniques can be combined to first extract clusters, which can then be extended to other dimensions of the folksonomy or to more elements of the same dimension by applying the FolkRank ranking scheme.*

The results in this chapter have been published as (Schmitz et al., 2006b) and (Hotho et al., 2006a).

11.1. Introduction

While the structure and user interface of folksonomies are sufficiently easy to use to attract large numbers of untrained users, there is still evidence that the envisioned goals of folksonomies as *social* bookmarking systems, namely, the formation of topical communities and the exploitation of communal knowledge, will not necessarily occur automatically and can benefit from support offered by the system:

> These findings [...] suggest that users require either longer time scales or better system supports to arrive at a globally coherent, navigable organization of resources by others.
>
> (Paolillo and Penumarthy, 2007)

While the previous chapter was about the global structural properties of the folksonomy graph, in this chapter we will make use of data mining and information retrieval techniques in order to support users when using a folksonomy system. We will present methods for ranking search results in Section 11.2 and for generating recommendations and detecting topical clusters using rule mining in Section 11.3.

11.2. Ranking in Folksonomies: *FolkRank*

The presentation of search results in an order that prefers the results most relevant to the querying user—the so-called *ranking*—is an integral part of modern information retrieval systems:

> This type of retrieval system [...] produces a list of records that "answer" the query, with the records ranked in order of likely relevance. Ranking retrieval systems are particularly appropriate for end-users.
>
> (Frakes and Baeza-Yates, 1992; p. 363)

Current folksonomy tools such as del.icio.us, however, provide only very limited searching support in addition to their browsing interface. Searching can be performed over tags and resource descriptions, but no ranking is done apart from ordering the hits in reverse chronological order.

In this section, we propose a ranking algorithm for folksonomies called *FolkRank*. It is inspired by spectral ranking schemes such as PageRank (Page et al., 1998) and HITS Kleinberg (1999) adapted to the particular structure of folksonomies.

11.2.1. Ranking in Folksonomies using Adapted PageRank

Folksonomy contents can be searched textually using traditional information retrieval methods. However, as the documents consist of short text snippets only (usually a description, such as the web page title, and the tags themselves), ordinary ranking schemes such as TF/IDF are not

feasible (see the discussion at the end of this section for an experiment indicating that retrieval based on term frequency does not perform well on our folksonomy datasets).

As shown in Chapter 8, a folksonomy induces a graph structure which we will exploit for ranking in this section. Our *FolkRank* algorithm is inspired by the seminal PageRank algorithm. The PageRank weight-spreading approach cannot be applied directly on folksonomies because of the different nature of folksonomies compared to the web graph (undirected triadic hyperedges instead of directed binary edges). In the following we discuss how to overcome this problem.

Adaptation of PageRank

We implement the weight-spreading ranking scheme on folksonomies in two steps. First, we transform the hypergraph between the sets of users, tags, and resources into an undirected, weighted, tripartite graph. On this graph, we apply a version of PageRank that takes into account the edge weights.

Converting the Folksonomy into an Undirected Graph. First we convert the folksonomy $\mathbb{F} = (U, T, R, Y)$ into an *un*directed tripartite graph $\mathbb{G}_{\mathbb{F}} = (V, E)$ as follows.

1. The set V of nodes of the graph consists of the disjoint union of the sets of tags, users and resources: $V = U \dot{\cup} T \dot{\cup} R$. (The tripartite structure of the graph can be exploited later for an efficient storage of the—sparse—adjacency matrix and the implementation of the weight-spreading iteration in the FolkRank algorithm.)

2. All cooccurrences of tags and users, users and resources, tags and resources become undirected, weighted edges between the respective nodes: $E = \{\{u,t\}, \{t,r\}, \{u,r\} \mid (u,t,r) \in Y\}$, with each edge $\{u,t\}$ being weighted with $|\{r \in R : (u,t,r) \in Y\}|$, each edge $\{t,r\}$ with $|\{u \in U : (u,t,r) \in Y\}|$, and each edge $\{u,r\}$ with $|\{t \in T : (u,t,r) \in Y\}|$.

   This way, a connection between, say, a user and a tag becomes more important the more resources occur with this (user, tag) pair.

Folksonomy-Adapted PageRank. The original formulation of PageRank (Brin and Page, 1998) reflects the idea that a web page is important if

there many pages linking to it, and if those pages are important themselves. The distribution of weights can thus be described as the fixed point of a weight passing scheme on the web graph, or equivalently, as the eigenvector of the largest eigenvalue 1 of a row-stochastic version of the adjacency matrix of the web graph.

This idea was extended in a similar fashion to bipartite subgraphs of the web in HITS (Kleinberg, 1999) and to n-ary directed graphs in (Xi et al., 2004). We employ the same underlying principle for our ranking scheme in folksonomies. The basic notion is that a resource which is tagged with important tags by important users becomes important itself. The same holds, symmetrically, for tags and users. Thus we have a graph of vertices which are mutually reinforcing each other by spreading their weights.

Like PageRank, we employ the random surfer model, a notion of importance for web pages that is based on the idea that an idealized random web surfer normally follows hyperlinks, but from time to time randomly jumps to a new web page without following a link. This results in the following definition of the rank of the vertices of the graph the entries in the fixed point $\vec{w}$ of the weight spreading computation $\vec{w} \leftarrow dA\vec{w} + (1-d)\vec{p}$, where $\vec{w}$ is a weight vector with one entry for each web page, A is the row-stochastic[1] version of the adjacency matrix of the graph $G_{\mathbb{F}}$ defined above, $\vec{p}$ is the random surfer component, and $d \in [0,1]$ is determining the influence of $\vec{p}$. In the original PageRank, $\vec{p}$ is used to outweigh the loss of weight on web pages without outgoing links. Usually, one will choose $\vec{p} = \mathbf{1} := (1,\ldots,1)^T$. In order to compute personalized PageRanks, however, $\vec{p}$ can be used to express user preferences by giving a higher weight to the components which represent the user's preferred web pages.

Formally, we spread the weight as follows:

$$\vec{w} \quad \leftarrow \quad \alpha\vec{w} + \beta A\vec{w} + \gamma\vec{p} \tag{11.1}$$

where A is the row-stochastic version of the adjacency matrix of $\mathbb{G}_{\mathbb{F}}$, $\vec{p}$ is a preference vector, $\alpha, \beta, \gamma \in [0,1]$ are constants with $\alpha + \beta + \gamma = 1$. The constant α is intended to regulate the speed of convergence, while the proportion between β and γ controls the influence of the preference vector.

We call the iteration to convergence according to Equation 11.1 the *Adapted PageRank* algorithm. Note that if $||\vec{w}||_1 = ||\vec{p}||_1$ holds and there

[1] I. e., each row of the matrix is normalized to 1 in the 1-norm.

are no rank sinks—the latter holds because the our graphs are undirected—the sum of the weights in the system will remain constant. The influence of different settings of the parameters α, β, and γ is discussed below.

Results for Adapted PageRank

We have evaluated the Adapted PageRank on the del.ico.us dataset described in Section 9.1. As there exists no "gold standard ranking" on these data, we evaluate our results empirically.

First, we studied the speed of convergence. We let $\vec{p} := \mathbf{1}$ (the vector having 1 in all components), and varied the parameter settings. In all settings, we discovered that $\alpha \neq 0$ slows down the convergence rate, but yields—of course—identical results. For instance, for $\alpha = 0.35, \beta = 0.65, \gamma = 0$, 411 iterations were needed, while $\alpha = 0, \beta = 1, \gamma = 0$ returned the same result in only 320 iterations. It turns out that using γ as a damping factor by spreading equal weight to each node in each iteration speeds up the convergence considerably by a factory of approximately 10 (e. g., 39 iterations for $\alpha = 0$, $\beta = 0.85$, $\gamma = 0.15$).

Tables 11.1, 11.2, and 11.3 show the result of the adapted PageRank algorithm for the 20 most important tags, users and resources, resp., computed with the parameters $\alpha = 0.35, \beta = 0.65, \gamma = 0$ (which equals the result for $\alpha = 0, \beta = 1, \gamma = 0$). Tags get the highest ranks, followed by the users, and the resources. Therefore, we present their rankings in separate lists.

As we can see from the tag table, the most important tag is "system:unfiled" which is used to indicate that a user did not assign any tag to a resource. It is followed by "web", "blog", "design" etc. This corresponds almost to the rank of the tags given by the overall tag count in the dataset.

The reason is that the graph $G_{\mathbb{F}}$ is undirected. We face thus the problem that, in the Adapted PageRank algorithm, weights that flow in one direction of an edge will basically 'swash back' along the same edge in the next iteration. Therefore the resulting is very similar (although not equal) to a ranking based on counting edge degrees. In fact, the ranking for an undirected graph without any damping factor, i. e., $\gamma = 0$, is the same as the normalized degree sequence of the graph—it is easy to prove that the normalized degree sequence is an eigenvector to the eigenvalue 1 of the row-stochastic adjacency matrix of the graph.

The resource ranking shows that Web 2.0 web sites like Slashdot, Wikipedia, Flickr, and a del.icio.us related blog appear in top positions. This

Table 11.1.: Folksonomy Adapted PageRank applied without preferences (called *baseline*) on tags

Tag	ad. PageRank
system:unfiled	0,0078404
web	0,0044031
blog	0,0042003
design	0,0041828
software	0,0038904
music	0,0037273
programming	0,0037100
css	0,0030766
reference	0,0026019
linux	0,0024779
tools	0,0024147
news	0,0023611
art	0,0023358
blogs	0,0021035
politics	0,0019371
java	0,0018757
javascript	0,0017610
mac	0,0017252
games	0,0015801
photography	0,0015469
fun	0,0015296

is not surprising, as early users of del.ico.us are likely to be interested in Web 2.0 related sites. This ranking correlates also strongly with a ranking based on degrees. The results for the top users are of more interest as different kinds of users appear. All top users have more than 6000 bookmarks; "notmuch" has a large amount of tags, while the tag count of "fritz" is considerably smaller.

To see how good the topic-specific ranking by Adapted PageRank works, we combined it with term frequency, a standard information retrieval weighting scheme. To this end, we downloaded all 3 million web pages referred to by a URL in our dataset. From these, we considered all plain text and HTML web pages, which left 2.834.801 documents. We converted all web pages into ASCII and computed an inverted index. To search for a term as in a search engine, we retrieved all pages containing the search term and ranked them by $tf(t) \cdot \vec{w}[v]$ where $tf(t)$ is the term frequency of search term t in page v, and $\vec{w}[v]$ is the Adapted PageRank weight of v.

Although this is a rather straightforward combination of two successful retrieval techniques, our experiments with different topic-specific queries indicate that this adaptation of PageRank does not work very well. For instance, for the search term "football", the del.icio.us home-

Table 11.2.: Folksonomy Adapted PageRank applied without preferences (called *baseline*) on users

User	ad. PageRank
shankar	0,0007389
notmuch	0,0007379
fritz	0,0006796
ubi.quito.us	0,0006171
weev	0,0005044
kof2002	0,0004885
ukquake	0,0004844
gearhead	0,0004820
angusf	0,0004797
johncollins	0,0004668
mshook	0,0004556
frizzlebiscuit	0,0004543
rafaspol	0,0004535
xiombarg	0,0004520
tidesonar02	0,0004355
cyrusnews	0,0003829
bldurling	0,0003727
onpause_tv_anytime	0,0003600
cataracte	0,0003462
triple_entendre	0,0003419
kayodeok	0,0003407

page showed up as the first result. Indeed, most of the highly ranked pages have nothing to do with football.

Other search terms provided similar results. Apparently, the overall structure of the—undirected—graph overrules the influence of the preference vector. In the next section, we discuss how to overcome this problem.

11.2.2. FolkRank—Topic-Specific Ranking in Folksonomies

In order to reasonably focus the ranking around the topics defined in the preference vector, we have developed a differential approach, which compares the resulting rankings with and without preference vector. This resulted in our new *FolkRank* algorithm.

The FolkRank Algorithm

The FolkRank algorithm computes a topic-specific ranking in a folksonomy as follows:

1. The preference vector $\vec{p}$ is used to determine the topic. It may have any distribution of weights, as long as $||\vec{w}||_1 = ||\vec{p}||_1$ holds. Typically a single entry or a small set of entries is set to a high value,

Table 11.3.: Folksonomy Adapted PageRank applied without preferences (called *baseline*) on resources

URL	ad. PageRank
http://slashdot.org/	0,0002613
http://pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html	0,0002320
http://script.aculo.us/	0,0001770
http://www.adaptivepath.com/publications/essays/archives/000385.php	0,0001654
http://johnvey.com/features/deliciousdirector/	0,0001593
http://en.wikipedia.org/wiki/Main_Page	0,0001407
http://www.flickr.com/	0,0001376
http://www.goodfonts.org/	0,0001349
http://www.43folders.com/	0,0001160
http://www.csszengarden.com/	0,0001149
http://wellstyled.com/tools/colorscheme2/index-en.html	0,0001108
http://pro.html.it/esempio/nifty/	0,0001070
http://www.alistapart.com/	0,0001059
http://postsecret.blogspot.com/	0,0001058
http://www.beelerspace.com/index.php?p=890	0,0001035
http://www.techsupportalert.com/best_46_free_utilities.htm	0,0001034
http://www.alvit.de/web-dev/	0,0001020
http://www.technorati.com/	0,0001015
http://www.lifehacker.com/	0,0001009
http://www.lucazappa.com/brilliantMaker/buttonImage.php	0,0000992
http://www.engadget.com/	0,0000984

and the remaining weight is equally distributed over the other entries. Since the structure of folksonomies is symmetric, we can define a topic by assigning a high value to either one or more tags and / or one or more users and / or one or more resources.

2. Let $\vec{w_0}$ be the fixed point from Equation (11.1) with $\gamma = 0$.

3. Let $\vec{w_1}$ be the fixed point from Equation (11.1) with $\gamma > 0$.

4. $\vec{w} := \vec{w_1} - \vec{w_0}$ is the final weight vector.

Thus, we compute the winners and losers of the mutual reinforcement of resources when a user preference is given, compared to the baseline without a preference vector. We call the resulting weight $\vec{w}[x]$ of an element x of the folksonomy the *FolkRank* of x.

Whereas the Adapted PageRank provides one global ranking, independent of any preferences, FolkRank provides one topic-specific ranking for each given preference vector. Note that a topic can be defined in the preference vector not only by assigning higher weights to specific tags, but also to specific resources and users. These three dimensions can even be combined in a mixed vector. Similarly, the ranking is not restricted to resources, it may as well be applied to tags and to users.

We will show below that indeed the rankings on all three dimensions provide interesting insights.

Comparing FolkRank with Adapted PageRank

To analyse the proposed FolkRank algorithm, we generated rankings for several topics, and compared them with the ones obtained from Adapted PageRank. We will here discuss two sets of search results, one for the tag "boomerang", and one for the URL http.//www.semanticweb.org. Our other experiments all provided qualitatively similar results.

Table 11.4 contains the ranked list of tags according to their weights from the Adapted PageRank by using the parameters $\alpha = 0.2, \beta = 0.5, \gamma = 0.3$, and 5 as a weight for the tag "boomerang" in the preference vector $\vec{p}$, while the other elements were given a weight of 0. As expected, the tag "boomerang" holds the first position while tags like "shop" or "wood" which are related are also under the top 20. The tags "software", "java", "programming" or "web", however, are on positions 4 to 7, but have nothing to do with "boomerang". The only reason for their showing up is that they are frequently used in del.icio.us (cf. Table 11.1). Table 11.5 contains the results of our FolkRank algorithm, again for the tag "boomerang". Intuitively, this ranking is better, as the globally frequent words disappear and related words like "wood" and "construction" are ranked higher.

A closer look reveals that this ranking still contains some unexpected tags; "kassel" or "rdf" are for instance not obviously related to "boomerang". An analysis of the user ranking (not displayed) explains this fact. The top-ranked user is "schm4704" (which is the del.icio.us username of the author), and he has indeed many bookmarks about boomerangs. A FolkRank run with preference weight 5 for user "schm4704" shows his different interests, see Table 11.7. His main interest apparently is in boomerangs, but other topics show up as well. In particular, he has a strong relationship to the tags "kassel" and "rdf". When a community in del.ico.us is small (such as the boomerang community), a single user can thus provide a strong bridge to other communities, a phenomenon that is also observed in small social communities.

A comparison of the FolkRank ranking for user "schm4704" (Table 11.7) with the Adapted PageRank result for that user confirms the initial finding from above, that the Adapted PageRank ranking (Table 11.6) contains many globally frequent tags, while the FolkRank ranking provides more personal tags. While the differential nature of the FolkRank algorithm usually pushes down the globally frequent tags such as

Table 11.4.: Adapted Pagerank with preference on tag *boomerang* for tags

Tag	ad. PRank
boomerang	0,4036883
shop	0,0069058
lang:de	0,0050943
software	0,0016797
java	0,0016389
programming	0,0016296
web	0,0016043
reference	0,0014713
system:unfiled	0,0014199
wood	0,0012378
kassel	0,0011969
linux	0,0011442
construction	0,0011023
plans	0,0010226
network	0,0009460
rdf	0,0008506
css	0,0008266
design	0,0008248
delicious	0,0008097
injuries	0,0008087
pitching	0,0007999

Table 11.5.: FolkRank with preference on tag *boomerang* for tags

Tag	FolkRank
boomerang	0,4036867
shop	0,0066477
lang:de	0,0050860
wood	0,0012236
kassel	0,0011964
construction	0,0010828
plans	0,0010085
injuries	0,0008078
pitching	0,0007982
rdf	0,0006619
semantic	0,0006533
material	0,0006279
trifly	0,0005691
network	0,0005568
webring	0,0005552
sna	0,0005073
socialnetworkanalysis	0,0004822
cinema	0,0004726
erie	0,0004525
riparian	0,0004467
erosion	0,0004425

Table 11.6.: Adapted Pagerank with preference on user *schm4704* for tags

Tag	ad. PRank
boomerang	0,0093549
lang:ade	0,0068111
shop	0,0052600
java	0,0052050
web	0,0049360
programming	0,0037894
software	0,0035000
network	0,0032882
kassel	0,0032228
reference	0,0030699
rdf	0,0030645
delicious	0,0030492
system:unfiled	0,0029393
linux	0,0029393
wood	0,0028589
database	0,0026931
semantic	0,0025460
css	0,0024577
social	0,0021969
webdesign	0,0020650
computing	0,0020143

Table 11.7.: FolkRank with preference on user *schm4704* for tags

Tag	FolkRank
boomerang	0,0093533
lang:de	0,0068028
shop	0,0050019
java	0,0033293
kassel	0,0032223
network	0,0028990
rdf	0,0028758
wood	0,0028447
delicious	0,0026345
semantic	0,0024736
database	0,0023571
guitar	0,0018619
computing	0,0018404
cinema	0,0017537
lessons	0,0017273
social	0,0016950
documentation	0,0016182
scientific	0,0014686
filesystem	0,0014212
userspace	0,0013490
library	0,0012398

Table 11.8.: FolkRank with preference on tag *boomerang* for resources

Url	FolkRank
http://www.flight-toys.com/boomerangs.htm	0,0047322
http://www.flight-toys.com/	0,0047322
http://www.bumerangclub.de/	0,0045785
http://www.bumerangfibel.de/	0,0045781
http://www.kutek.net/trifly_mods.php	0,0032643
http://www.rediboom.de/	0,0032126
http://www.bws-buhmann.de/	0,0032126
http://www.akspiele.de/	0,0031813
http://www.medco-athletics.com/education/elbow_shoulder_injuries/	0,0031606
http://www.sportsprolo.com/sports%20prolotherapy%20newsletter%20pitching%20injuries.htm	0,0031606
http://www.boomerangpassion.com/english.php	0,0031005
http://www.kuhara.de/bumerangschule/	0,0030935
http://www.bumerangs.de/	0,0030935
http://s.webring.com/hub?ring=boomerang	0,0030895
http://www.kutek.net/boomplans/plans.php	0,0030873
http://www.geocities.com/cmorris32839/jonas_article/	0,0030871
http://www.theboomerangman.com/	0,0030868
http://www.boomerangs.com/index.html	0,0030867
http://www.lmifox.com/us/boom/index-uk.htm	0,0030867
http://www.sports-boomerangs.com/	0,0030867
http://www.rangsboomerangs.com/	0,0030867

Table 11.9.: Folkrank with preference on user *schm4704* for resources

Url	FolkRank
http://jena.sourceforge.net/	0,0019369
http://www.openrdf.org/doc/users/ch06.html	0,0017312
http://dsd.lbl.gov/~hoschek/colt/api/overview-summary.html	0,0016777
http://librdf.org/	0,0014402
http://www.hpl.hp.com/semweb/jena2.htm	0,0014326
http://jakarta.apache.org/commons/collections/	0,0014203
http://www.aktors.org/technologies/ontocopi/	0,0012839
http://eventseer.idi.ntnu.no/	0,0012734
http://tangra.si.umich.edu/~radev/	0,0012685
http://www.cs.umass.edu/~mccallum/	0,0012091
http://www.w3.org/TR/rdf-sparql-query/	0,0011945
http://ourworld.compuserve.com/homepages/graeme_birchall/HTM_COOK.HTM	0,0011930
http://www.emory.edu/EDUCATION/mfp/Kuhn.html	0,0011880
http://www.hpl.hp.com/semweb/rdql.htm	0,0011860
http://jena.sourceforge.net/javadoc/index.html	0,0011860
http://www.geocities.com/mailsoftware42/db/	0,0011838
http://www.quirksmode.org/	0,0011327
http://www.kde.cs.uni-kassel.de/lehre/ss2005/googlespam	0,0011110
http://www.powerpage.org/cgi-bin/WebObjects/powerpage.woa/wa/story?newsID=14732	0,0010402
http://www.vaughns-1-pagers.com/internet/google-ranking-factors.htm	0,0010329
http://www.cl.cam.ac.uk/Research/SRG/netos/xen/	0,0010326

"web", though, this happens in a differentiated manner: FolkRank will keep them in the top positions, *if* they are indeed relevant to the user under consideration. This can be seen for example for the tags "web" and "java". While the tag "web" appears in schm4704's tag list—but not very often, "java" is a very important tag for that user. This is reflected in the FolkRank ranking: "java" remains in the top 5, while "web" is pushed down in the ranking.

The ranking of the resources for the tag "boomerang" given in Table 11.8 also provides interesting insights. As shown in the table, many boomerang related web pages show up. Comparing the top 20 web pages of "boomerang" with the top 20 pages given by the "schm4704" ranking, there is no "boomerang" web page in the latter. This can be explained by analysing the tag distribution of this user. While "boomerang" is the most frequent tag for this user, in del.icio.us, "boomerang" appears rather infrequently. The first boomerang web page in the "schm4704" ranking is the 21st URL (i. e., just outside the listed top 20). Thus, while the tag "boomerang" itself dominates the tags of this user, in the whole, the semantic web related tags and resources prevail. This demonstrates that while the user "schm4704" and the tag "boomerang" are strongly correlated, we can still get an overview of the respective related items which shows several topics of interest for the user.

Let us consider a second example. Tables 11.10 through 11.14 present the results for preference on the web page http://www.semanticweb.org/. Tables 11.10 and 11.11 show the Adapted PageRank for tags and users, resp., and Tables 11.12 and 11.13 show the FolkRank results.

Again, we see that the differential ranking of FolkRank makes the right decisions: in the Adaptive PageRank, globally frequent tags such as "web", "css", "xml", "programming" get high ranks. Of these, only two turn up to be of genuine interest to the members of the Semantic Web community: "web" and "xml" remain at high positions, while "css" and "programming" disappear altogether from the list of the 20 highest ranked tags. Also, several variations of tags which are used to label Semantic Web related pages appear (or get ranked higher): "semantic web" (two tags, space-separated), "semantic_web", "semweb", "sem-web". These cooccurrences of similar tags could be exploited further to consolidate the emergent semantics of a field of interest. While the discovery in this case may also be done in a simple syntactic analysis, the graph based approach allows also for detecting inter-community and inter-language relations.

The user IDs can not be checked for topical relatedness immediately, since they are not related to the users' full names—although a former

Table 11.10.: Adapted pagerank for tags with preference on resource http://www.semanticweb.org/

Tag	ad. PRank
semanticweb	0,0208605
web	0,0162033
semantic	0,0122028
system:unfiled	0,0088625
semantic_web	0,0072150
rdf	0,0046348
semweb	0,0039897
resources	0,0037884
community	0,0037256
xml	0,0031494
research	0,0026720
programming	0,0025717
css	0,0025290
portal	0,0024118
.imported	0,0020495
imported-bo...	0,0019610
en	0,0018900
science	0,0018166
.idate2005-04-11	0,0017779
newfurl	0,0017578
internet	0,0016122

winner of the Semantic Web Challenge and the best paper award at a Semantic Web Conference seems to be among them. The web pages that appear in the top list, on the other hand, include many well-known resources from the Semantic Web area. An interesting resource on the list of top-rated resources, presented in Table 11.14, is PiggyBank, which has been presented in November 2005 at the ISWC conference. Considering that the dataset was crawled in July 2005, when PiggyBank was not that well known, the prominent position of PiggyBank in del.icio.us at such an early time is an interesting result. This indicates the sensibility of social bookmarking systems for upcoming topics.

These two examples—as well as the other experiments we performed—show that FolkRank provides good results when querying the folksonomy for topically related elements. Overall, our experiments indicate that topically related items can be retrieved with FolkRank for any given set of highlighted tags, users and / or resources.

Our results also show that the current size of folksonomies is still prone to being skewed by a relatively small number of perturbations – a single user, at the moment, can influence the emergent understanding of a certain topic in the case that a sufficient number of different points of view for such a topic has not been collected yet. With the growth of folksonomy-based data collections on the web, the influence of single

Table 11.11.: Adapted pagerank for users with preference on resource http://www.semanticweb.org/

User	ad. PageRank
up4	0,0091995
awenger	0,0086261
j.deville	0,0074021
chaizzilla	0,0062570
elektron	0,0059457
captsolo	0,0055671
stevag	0,0049923
dissipative	0,0049647
krudd	0,0047574
williamteo	0,0037204
stevecassidy	0,0035887
pmika	0,0035359
millette	0,0033028
myren	0,0028117
morningboat	0,0025913
philip.fennell	0,0025338
mote	0,0025212
dnaboy76	0,0024813
webb.	0,0024709
nymetbarton	0,0023790
alphajuliet	0,0023781

users will fade in favor of a common understanding provided by huge numbers of users.

As detailed above, our ranking is based on tags only, without regarding any inherent features of the resources at hand. This allows to apply FolkRank to search for pictures (e. g., in Flickr) and other multimedia content, as well as for all other items that are difficult to search in a content-based fashion. The same holds for intranet applications, where in spite of centralized knowledge management efforts, documents often remain unused because they are not hyperlinked and thus difficult to find.

Generating Recommendations

The original PageRank paper (Brin and Page, 1998) already pointed out the possibility of using the random surfer vector $\vec{p}$ as a personalization mechanism for PageRank computations. The results of Section 11.2.2 show that, given a user, one can find set of tags and resources of interest to him. Likewise, FolkRank yields a set of related users and resources for a given tag. Following these observations, FolkRank can be used to generate recommendations within a folksonomy system. These recom-

Table 11.12.: FolkRank for tags with preference on resource http://www.semanticweb.org/

Tag	FolkRank
semanticweb	0,0207820
semantic	0,0121305
web	0,0118002
semantic_web	0,0071933
rdf	0,0044461
semweb	0,0039308
resources	0,0034209
community	0,0033208
portal	0,0022745
xml	0,0022074
research	0,0020378
imported-bo...	0,0018920
en	0,0018536
.idate2005-04-11	0,0017555
newfurl	0,0017153
tosort	0,0014486
cs	0,0014002
academe	0,0013822
rfid	0,0013456
sem-web	0,0013316
w3c	0,0012994

mendations can be presented to the user at different points in the usage of a folksonomy system:

- Documents that are of potential interest to a user can be suggested to him. This kind of recommendation pushes potentially useful content to the user and increases the chance that a user finds useful resources that he did not even know existed by “serendipitous” browsing.
- When using a certain tag, other related tags can be suggested. This can be used, for instance, to speed up the consolidation of different terminologies and thus facilitate the emergence of a common vocabulary.
- While folksonomy tools already use simple techniques for tag recommendations, FolkRank additionally considers the tagging behavior of other users.
- Other users that work on related topics can be made explicit, improving thus the knowledge transfer within organizations and fostering the formation of communities.

Table 11.13.: FolkRank for users with preference on resource http://www.semanticweb.org/

User	FolkRank
up4	0,0091828
awenger	0,0084958
j.deville	0,0073525
chaizzilla	0,0062227
elektron	0,0059403
captsolo	0,0055369
dissipative	0,0049619
stevag	0,0049590
krudd	0,0047005
williamteo	0,0037181
stevecassidy	0,0035840
pmika	0,0035358
millette	0,0032103
myren	0,0027965
morningboat	0,0025875
philip.fennell	0,0025145
webb.	0,0024671
dnaboy76	0,0024659
mote	0,0024214
alphajuliet	0,0023668
nymetbarton	0,0023666

At the end of the following section, we will show how the FolkRank ranking can be combined with other algorithms to find and extend topical clusters in folksonomies.

11.3. Mining Association Rules in Folksonomies

The previous section has presented a method of ranking tags, users, and resources in order to select the most relevant elements of a folksonomy concerning given user preferences.

A related, though different problem we discussed in Section 7.3 is to allow the user to recognize and make use of the structure inherent in the folksonomy, and to add structure himself, e. g., by populating his $\prec$ relation.

A first step towards more structure within folksonomy systems is to discover knowledge that is already implicitly present by the way different users assign tags to resources. This knowledge may be used for recommending both a hierarchy on the already existing tags, and additional tags, ultimately leading towards *emergent semantics* (Steels, 1998; Staab et al., 2002) by converging use of the same vocabulary. In this

Table 11.14.: FolkRank for resources with preference on resource http://www.semanticweb.org/

URL	FolkRank
http://www.semanticweb.org/	0,3761957
http://flink.semanticweb.org/	0,0005566
http://simile.mit.edu/piggy-bank/	0,0003828
http://www.w3.org/2001/sw/	0,0003216
http://infomesh.net/2001/swintro/	0,0002162
http://del.icio.us/register	0,0001745
http://mspace.ecs.soton.ac.uk/	0,0001712
http://www.adaptivepath.com/publications/essays/archives/000385.php	0,0001637
http://www.ontoweb.org/	0,0001617
http://www.aaai.org/AITopics/html/ontol.html	0,0001613
http://simile.mit.edu/	0,0001395
http://itip.evcc.jp/itipwiki/	0,0001256
http://www.google.be/	0,0001224
http://www.letterjames.de/index.html	0,0001224
http://www.daml.org/	0,0001216
http://shirky.com/writings/ontology_overrated.html	0,0001195
http://jena.sourceforge.net/	0,0001167
http://www.alistapart.com/	0,0001102
http://www.federalconcierge.com/WritingBusinessCases.html	0,0001060
http://pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html	0,0001059
http://www.shirky.com/writings/semantic_syllogism.html	0,0001052

sense, knowledge discovery (KDD) techniques are a promising tool for bottom-up building of conceptual structures.

In this section, we will focus on a particular KDD technique, namely association rules. Since folksonomies provide a three-dimensional dataset (users, tags, and resources) instead of a usual two-dimensional one (items and transactions), we present first a systematic overview of projections of a folksonomy onto a two-dimensional structure. Then we will show the results of mining rules from two selected projections on the del.icio.us system.

11.3.1. Association Rule Mining

We assume here that the reader is familiar with the basics of association rule mining introduced by Agrawal et al. (1993). As the work presented in this section is on the conceptual rather than on the computational level, we refrain in particular from describing the vast area of developing efficient algorithms. Many of the existing algorithms can be found at the Frequent Itemset Mining Implementations Repository.[2] Instead, we just recall the definition of the association rule mining problem, which was initially stated by Agrawal et al. (1993), in order to clarify the no-

[2] http://fimi.cs.helsinki.fi/

tations used in the following. We will not follow the original terminology of Agrawal et al., but rather use the vocabulary of Formal Concept Analysis (FCA) (Wille, 1982),[3] as it better fits with the formal folksonomy model introduced in Chapter 8. The definition of the rule mining problem in FCA terminology follows Stumme (2002a).

Definition 2. *A* formal context *is a dataset* $\mathbb{K} := (G, M, I)$ *consisting of a set* G *of* objects*, a set* M *of* attributes*, and a binary relation* $I \subseteq G \times M$*, where* $(g, m) \in I$ *is read as "object* g *has attribute* m*".*

In the usual basket analysis scenario, M is the set of items sold by a supermarket, G is the set of all transactions, and, for a given transaction $g \in G$, the set $g^I := \{m \in M | (g, m) \in I\}$ contains all items bought in that transaction.

Definition 3 (Derivation Operator, Support)**.** *For a set* A *of attributes, we define* $A' := \{g \in G \mid \forall m \in A\colon (g, m) \in I\}$*. The* support *of* A *is calculated by* $\text{supp}(A) := \frac{|A'|}{|G|}$.

Definition 4 (Association Rule Mining Problem (Agrawal et al., 1993))**.** *Let* $\mathbb{K}$ *be a formal context, and* $\text{minsupp}, \text{minconf} \in [0, 1]$*, called* minimum support *and* minimum confidence thresholds*, resp. The* association rule mining problem *consists now of determining all pairs* $A \to B$ *of subsets of* M *whose* support $\text{supp}(A \to B) := \text{supp}(A \cup B)$ *is above the threshold* minsupp*, and whose* confidence $\text{conf}(A \to B) := \frac{supp(A \cup B)}{supp(A)}$ *is above the threshold* minconf.

As the rules $A \to B$ and $A \to B \setminus A$ carry the same information, and in particular have same support and same confidence, we will consider in this section the additional constraint prevalent in the data mining community, that premise A and conclusion B are to be disjoint.[4]

When comparing Definition 1 on page 107 and Definition 2, we observe that association rules cannot be mined directly on folksonomies due to their triadic nature. One either has to define some kind of triadic association rules, or to transform the triadic folksonomy into a dyadic formal context. In this section, we follow the latter approach.

[3] For a detailed discussion about the role of FCA for association rule mining see (Stumme, 2002a).

[4] In contrast, in FCA, one often requires A to be a subset of B, as this fits better with the notion of *closed itemsets* which arose of applying FCA to the association mining problem (Pasquier et al., 1999; Zaki and Hsiao, 1999; Stumme, 1999).

11.3.2. Projecting the Folksonomy onto two Dimensions

As discussed in the previous section, we have to reduce the three-dimensional folksonomy to a two-dimensional formal context before we can apply any association rule mining technique. Several such projections have already been introduced by Lehmann and Wille (1995). Stumme (2005) provides a more complete approach, which we will adapt to the association rule mining scenario.

As we want to analyze all facets of the folksonomy, we want to allow to use any of the three sets U, T, and R as the set of objects —on which the support is computed—at some point in time, depending on the task on hand. Therefore, we will not fix the roles of the three sets in advance. Instead, we consider a triadic context as a symmetric structure, where all three sets are of equal importance. For easier handling, we will therefore denote the folksonomy $\mathbb{F} := (U, T, R, Y)$ alternatively by $\mathbb{F} := (X_1, X_2, X_3, Y)$ in the following.

We will define the set of objects—i. e., the set on which the support will be counted—by a permutation on the set $\{1, 2, 3\}$, i. e., by an element σ of the full symmetric group S_3. The choice of a permutation indicates, together with one of the aggregation modes '$\mathfrak{Ɔ}$', '$\mathfrak{W}$', '$\exists n$' with $n \in \mathbb{N}$, and '$\forall$', on which formal context $\mathbb{K} := (G, M, I)$ the association rules are computed.

- $\mathbb{K}^{\sigma,\mathfrak{Ɔ}} := (X_{\sigma(1)} \times X_{\sigma(2)}, X_{\sigma(3)}, I)$ with $((x_{\sigma(1)}, x_{\sigma(2)}), x_{\sigma(3)}) \in I$ if and only if $(x_1, x_2, x_3) \in Y$.

  In this projection, pairs of elements become the objects of the context, while single elements are the attributes. The examples at the end of this section are of this type, e. g., considering *(user, tag)* pairs as objects and resources as attributes.

- $\mathbb{K}^{\sigma,\mathfrak{W}} := (X_{\sigma(1)}, X_{\sigma(2)} \times X_{\sigma(3)}, I)$ with $(x_{\sigma(1)}, (x_{\sigma(2)}, x_{\sigma(3)})) \in I$ if and only if $(x_1, x_2, x_3) \in Y$.

  In this projection, single elements of the folksonomy are objects, while pairs are attributes; e. g., one can compute association rules on sets of *(user, tag)* attributes on resources as objects.

- $\mathbb{K}^{\sigma,\exists n} := (X_{\sigma(1)}, X_{\sigma(2)}, I)$ with $(x_{\sigma(1)}, x_{\sigma(2)}) \in I$ if and only if there exist n different $x_{\sigma(3)} \in X_{\sigma(3)}$ with $(x_1, x_2, x_3) \in Y$.

  This a plain projection of Y to two dimensions $\sigma(1), \sigma(2)$ with the restriction that only those pairs occur in $\mathbb{K}^{\sigma,\exists n}$ for which there are at least n preimages in Y. An example would be to build a context

with resources as objects and tags as attributes, where only those pairs are considered that are posted by at least n users.

- $\mathbb{K}^{\sigma,\forall} := (X_{\sigma(1)}, X_{\sigma(2)}, I)$ with $(x_{\sigma(1)}, x_{\sigma(2)}) \in I$ if and only if for all $x_{\sigma(3)} \in X_{\sigma(3)}$ holds $(x_1, x_2, x_3) \in Y$. The mode '$\forall$' is thus equivalent to '$\exists n$' if $|X_{\sigma(3)}| = n$.

  This mode is listed here for completeness, though in the case of real-life folksonomies, it is unrealistic that this context will ever be non-empty. It would be in the case of $\sigma = (1, 2, 3)$, for example, iff there was a tag that was applied by a user to all resources in R.

These projections are complemented by the following way to 'cut slices' out of the folksonomy. A slice is obtained by selecting one dimension (out of user / tag / resource), and then fixing in this dimension one particular instance.

- For $x := x_{\sigma(3)} \in X_{\sigma(3)}$, $\mathbb{K}^{\sigma,x} := (X_{\sigma(1)}, X_{\sigma(2)}, I)$ with $(x_{\sigma(1)}, x_{\sigma(2)}) \in I$ if and only if $(x_1, x_2, x_3) \in Y$.

  An example for this slicing operation would be the context $\mathbb{K}^{(1,2,3),r_1}$ that contains the users as objects and the tags as attributes that occur in Y with a particular resource r_1, thus allowing for a closer examination of the tagging behavior on that resource. The same definition can be extended to larger slices if we do not consider a single $x \in X_{\sigma(3)}$, but a set $X \subseteq X_{\sigma 3}$ of elements of that dimension. This would allow, for example, to examine the tagging behavior on a group of resources.

In the next section, we will discuss for a selected subset of these projections the kind of rules one obtains from mining the formal context that is resulting from the projection.

11.3.3. Mining Association Rules on the Projected Folksonomy

After having performed one of the projections described in the previous section, one can now apply the standard association rule mining techniques as described in Section 11.3.1. For clarity of presentation, we will focus on a subset of projections.

In particular, we will address the two projections $\mathbb{K}^{\sigma_i,\flat}$ with $\sigma_1 := (1 \mapsto 1, 2 \mapsto 3, 3 \mapsto 2)$ and $\sigma_2 := \mathrm{id}$. We obtain the two dyadic contexts $\mathbb{K}_1 := (U \times R, T, I_1)$ with $I_1 := \{((u, r), t) | (u, t, r) \in Y\}$ and $\mathbb{K}_2 := (U \times T, R, I_2)$ with $I_2 := \{((u, t), r) | (u, t, r) \in Y\}$.

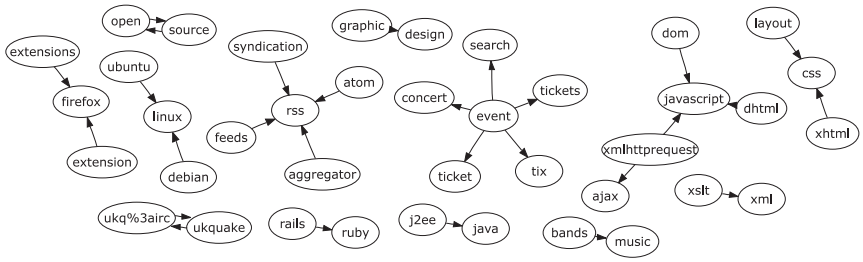

Figure 11.1.: All rules $A \rightarrow B$ with $|A| = |B| = 1$ of $\mathbb{K}_1$ with .05 % support, 50 % confidence

For computing the associations, we have used the implementation described in (Borgelt, 2004)[5].

Example: Association Rules between Tags

An association rule $A \rightarrow B$ in $\mathbb{K}_1$ is read as: users assigning the tags from a set A of tags to some resources often also assign the tags from B to these. This type of rules may be used in a recommender system. If a user assigns all tags from A then the system suggests that he might also want to add those from B.

Figure 11.1 shows all rules with one element in the premise and one element in the conclusion that we derived from $\mathbb{K}_1$ with a minimum support of 0.05 % and a minimum confidence of 50 %. In the diagram one can see that our interpretation of rules in $\mathbb{K}_1$ holds for these examples: users tagging some web page with *debian* are likely to tag it with *linux* also, and pages about *bands* are probably also concerned with *music*. These results can be used in a recommender system, aiding the user in choosing the tags which are most helpful in retrieving the resource later.

Another view on these rules is to see them as subsumption relations, so that the rule mining can be used to learn a taxonomic structure. If many resources tagged with *xslt* are also tagged with *xml*, this indicates, for example, that *xml* can be considered a supertopic of *xslt* if one wants to automatically populate the $\prec$ relation. Figure 11.1 also shows two pairs of tags which occur together very frequently without any distinct direction in the rule: *open source* occurs as a phrase most of the time, while the other pair consists of two tags (*ukquake* and *ukq:irc*), which we assume are added automatically to any resource that is mentioned in a particular chat channel.

[5] http://fuzzy.cs.uni-magdeburg.de/~borgelt/apriori.html

Example: Association Rules between Resources

The second example are association rules $A \rightarrow B$ in $\mathbb{K}_2$ which are read as: users labeling the resources in the set A of resources with some tags often also assign these tags to the resources in B. In essence both resources have to have something in common. Figure 11.2 shows parts of the resulting graph for applying association rules with 0.05 % support, and 10 % confidence on $\mathbb{K}_2$. Only associations rules with one element in premise and one element in conclusion are considered here. In Figure 11.2 we identified several clusters in the graph. In this case, the clusters were identified visually by considering connected components as well as spatial proximity in the layout generated by a spring embedder approach; more elaborate graph clustering algorithms could be employed for this step. Upon manual inspection, these clusters can easily be labelled with topics such as *delicious hacks*, *javascript*, *ajax*, or *photo collections*.

In the following, we will discuss some of the clusters in the graph of association rules depicted in Figure 11.2. Figures 11.3, 11.4, and 11.5 show some details from the graph. One of the most distinguished clusters in the graph is the subgraph in Figure 11.3. It is clearly separated from the rest of the graph and is obviously exclusively concerned with photo sharing.

A similar case is the graph in Figure 11.5, which is also a connected component in itself, but contains several topics: there are news pages such as *The Register*, *The New York Times* etc., and these are connected to pages about hacker productivity tools such as *Lifehacker* and *TiddlyWiki* through specialized computer-related news sites, *Wired* and *Slashdot*. This demonstrates that by evaluating tagging behavior, not only topical clusters themselves—consisting of densely connected tags, users, or resources—but also higher-level connections between clusters can be found.

11.3.4. Labeling and Fuzzy Extension of Clusters

As an example of how the folksonomy mining techniques from this section and Section 11.2 can be combined, we will demonstrate the labeling of resource clusters in the association rule graph. Consider the community depicted in Figure 11.6. It is obvious that this set of pages is concerned with an extension called *GreaseMonkey*[6] for the popular *Firefox*[7]

[6] ttp://greasemonkey.mozdev.org/
[7] http://www.mozilla.com/

web browser. This extension enables the user to apply custom JavaScript code to web pages she browses.

We applied the FolkRank algorithm from Section 11.2 with a preference vector adding weight to the four resources in Figure 11.6. As we are interested in labeling the cluster, we first consider the top tags from the resulting FolkRank vector. The top tags can be seen in Table 11.15. Obviously, these tags capture very well the contents of this cluster: it is indeed concerned with a *firefox extension* named *greasemonkey* that applies *javascript* to *web* pages, etc.

The graph derived from the association rules shows connections that are made with respect to sharply defined criteria: only those tags, users, or resources will be connected that meet the minimum support and confidence of the rule mining step. In order to extend the clusters found in those graphs to the other dimensions, and to add items that are closely related without meeting the association rule thresholds, one can apply the FolkRank algorithm to yield a fuzzy extension of clusters that were derived using rule mining. As an example consider Table 11.16: it shows the top related resources to the ones from the GreaseMonkey cluster as computed by the FolkRank algorithm. It can be seen that those resources cover additional Greasemonkey resources, as well as related resources on Firefox extensions as well as del.icio.us and Google hacks.

11.4. Related Work

The FolkRank algorithm makes use of the seminal PageRank algorithm introduced by Page et al. (1998). While several variations for bipartite graphs (Kleinberg, 1999), multilevel graphs (Xi et al., 2004), or object structures (Balmin et al., 2004; Chirita et al., 2006) have been proposed, none of them fits exactly for the folksonomy setting. A strategy for ranking ontology items based on observing changes in weight spreading schemes was proposed by Alani et al. (2003).

Suggestions for ranking in folksonomies include (Michail, 2005; Szekely and Torres, 2005). Both are follow the approach of ranking one dimension after the other, e. g., first ranking the importance of users and then ranking tags according to the users' weights.

Mining the structure of folksonomies has been one of the first interests of the scientific community in folksonomies. Mining approaches include the extraction of relationships in the tag set by exploiting tag cooccurrence (Mika, 2005; Schmitz, 2006; Heymann and Garcia-Molina,

Table 11.15.: Top related tags for the GreaseMonkey cluster

firefox
greasemonkey
web
javascript
extension
extensions
css
mozilla
programming
tools
software
webdev
webdesign
dhtml
design
scripts
html
reference
ajax

Table 11.16.: Top related resources for the GreaseMonkey cluster

http://dunck.us/collab/GreaseMonkeyUserScripts
http://diveintogreasemonkey.org/
http://greasemonkey.mozdev.org/
http://www.karmatics.com/aardvark/
http://dietrich.ganx4.com/foxylicious/
http://platypus.mozdev.org/
http://www.letitblog.com/greasemonkey-compiler/
http://diveintogreasemonkey.org/toc/
http://delicious.mozdev.org/
http://script.aculo.us/
http://www.nivi.com/blog/article/greasemonkey-and-business-models/
http://extensions.roachfiend.com/howto.php
http://pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html
http://ejohn.org/projects/autodelicious/
http://persistent.info/archives/2005/03/01/gmail-searches
http://roachfiend.com/archives/2004/12/08/how-to-create-firefox-extensions/
http://johnhaller.com/jh/mozilla/portable_firefox/
http://johnvey.com/features/deliciousdirector/
http://www.customizegoogle.com/

2006; Niwa et al., 2006) as well as trend detection on tags (Hotho et al., 2006b; Dubinko et al., 2006).

The rule mining approach chosen in this chapter is related to triadic Formal Concept Analysis (Lehmann and Wille, 1995; Stumme, 2005). We have continued working in that direction and devised a triadic mining algorithm for Iceberg tri-lattices that can be applied to folksonomies (Jäschke et al., 2006).

11.5. Conclusion and Outlook

11.5.1. Summary

In this chapter, we have presented two folksonomy mining approaches that can help to gain an understanding of the structure of a folksonomy, as well as provide concrete rankings and recommendations that can be presented to the user and support his browsing and searching activities when using a folksonomy tool.

The *FolkRank* ranking algorithm takes into account the graph structure of folksonomies in order to provide a ranking of tags, users, and resources based on user preferences. We have seen that the top folksonomy elements which are retrieved by FolkRank tend to fall into a coherent topic area, e.g. "Semantic Web". This leads naturally to the idea of extracting *communities of interest*[8] from the folksonomy, which are represented by their top tags and the most influential persons and resources. If these communities are made explicit, interested users can find them and participate, and community members can more easily get to know each other and learn of others' resources.

While the FolkRank algorithm can be used to compute fuzzy clusters around given preferences, the rule mining approach we demonstrated generated crisp sets of rules. By partitioning the folksonomy graph according to these rules, strongly coherent clusters can be identified. The combination of both techniques opens up an interesting possibility, e. g., by computing crisp clusters using rule mining, and using FolkRank to label or extend the clusters.

11.5.2. Outlook

Regarding the techniques presented in this chapter, several areas of research present themselves for future work:

[8] see (Maier, 2005; p. 160ff) for a detailed discussion of different kinds of communities

Trend Detection: The techniques presented in this chapter have been applied to datasets consisting of static snapshots of folksonomies so far. One interesting aspect which has been touched in Chapter 10 already is the development of folksonomies over time. In (Hotho et al., 2006b), we have used the FolkRank algorithm from this chapter to analyze the trends and changes in a folksonomy, highlighting, for example, tags that have gained or lost in popularity within a certain period of time. Another method for trend detection in folksonomies, though employing a simpler algorithm and more focused on scalability, was proposed by Dubinko et al. (2006).

Spam Detection: Judging from our own experience with running BibSonomy, as well as from the discussions of the mailing lists of the popular folksonomy systems such as del.icio.us and CiteULike, spamming is an increasing problem in folksonomies. One possible application of the ranking schemes presented in this section is the (semi-)automatic detection of spammers. We are currently exploring the possibilities in that direction.

Scalability: Even with efficient implementations, the algorithms presented in this chapter are not yet usable for generating recommendations or rankings online, i. e., at the time of user interaction. Thus, the only way of integrating these algorithms into a live system would be to process a batch of rankings or rule mining steps offline and present them the next time a user logs in.

Further research can explore several possible ways of making these algorithms more tractable in practice, either by improving the implementations as such, or by following lines of research similar to those that have been pursued in web ranking. One way of speeding up spectral ranking algorithms is by divide-and-conquer approaches (Jeh and Widom, 2003; Kamvar et al., 2003a,b) that compute global rankings from a number of smaller ranking problems. These optimizations could be transferred to the FolkRank algorithm.

Structuring Folksonomies Non-Intrusively: When folksonomy systems attract millions of users that contribute content, user support has to go beyond enhanced retrieval facilities. Thus, the internal structure of a folksonomy has to be improved, e. g., by adding relations between tags.

One approach would be to extend the folksonomy model towards Semantic Web technologies. The key question remains though how to exploit the benefits of a richer and more structured knowledge representation without bothering untrained users with its acquisition and maintenance effort and rigidity. We believe that this will become a fruitful research area for the Semantic Web and folksonomy communities for the next years.

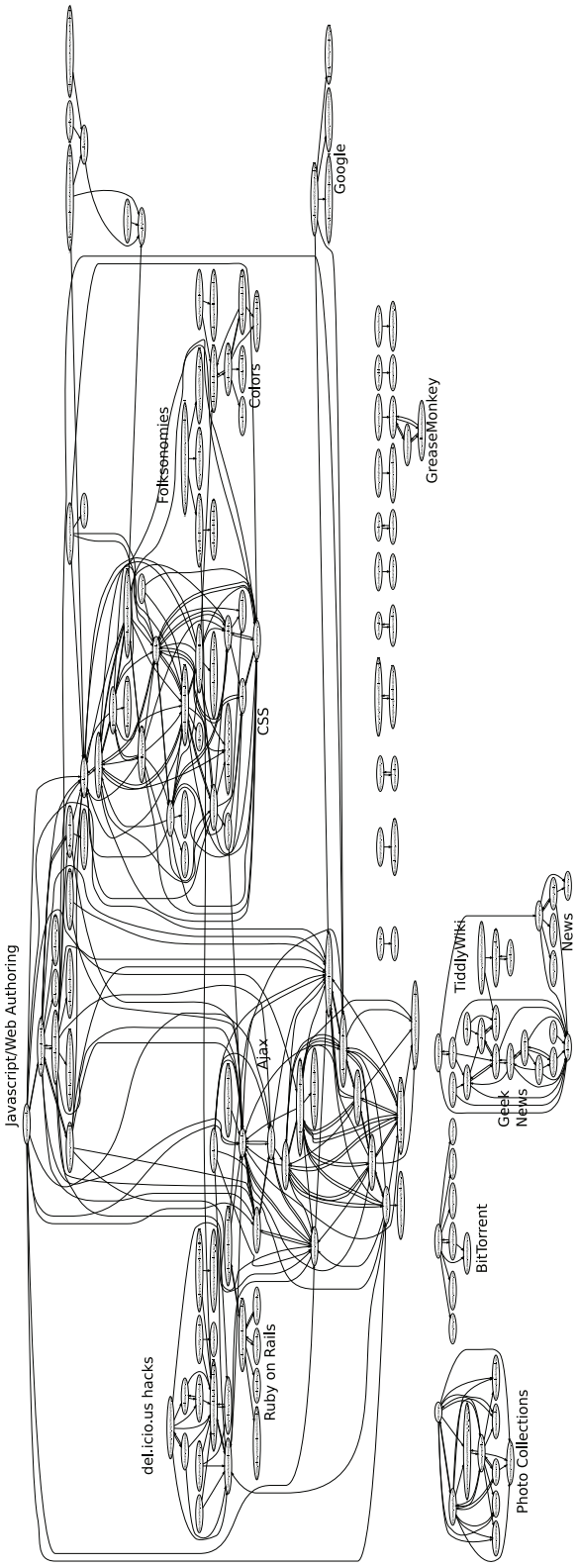

Figure 11.2.: All rules with two elements of $\mathbb{K}_2$ with 0.05 % support, and 10 % confidence

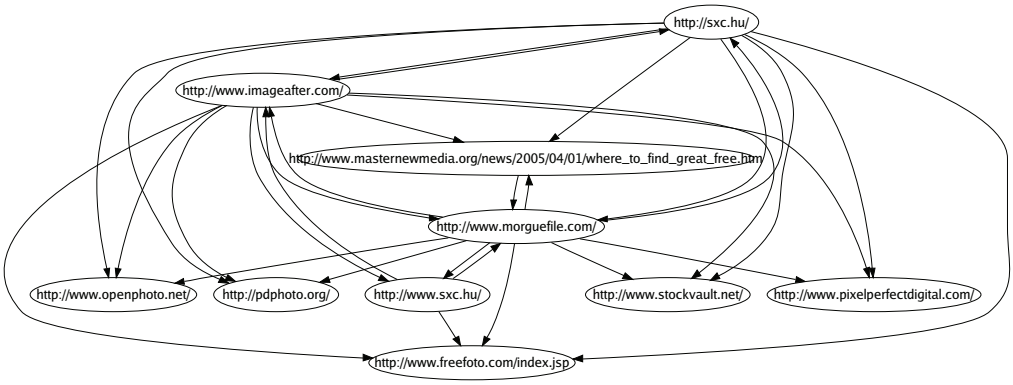

Figure 11.3.: Cluster within Figure 11.2: photo collections

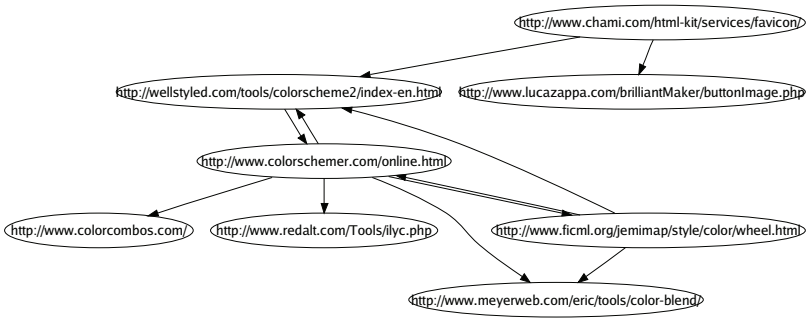

Figure 11.4.: Cluster within Figure 11.2: color schemes for web pages

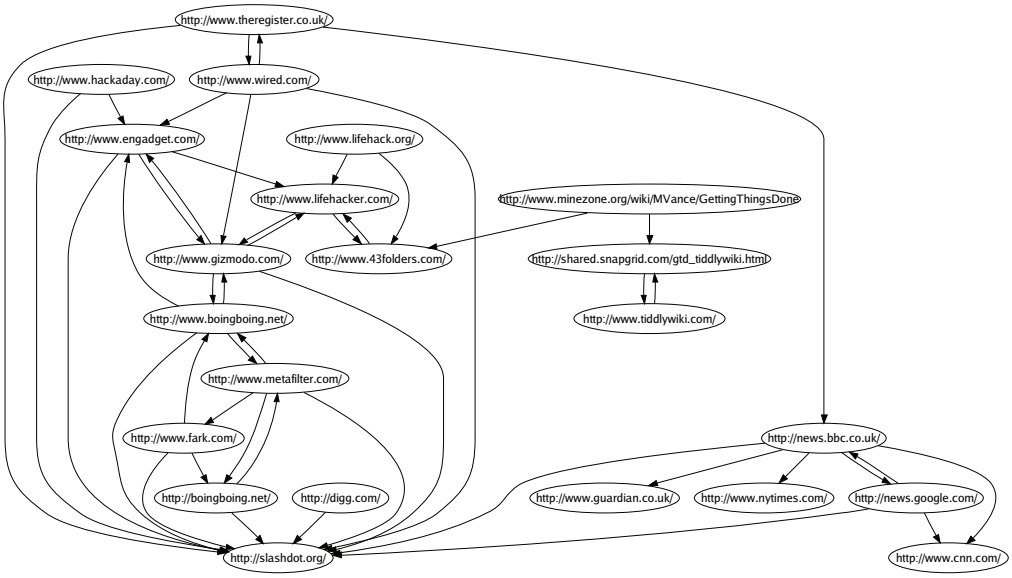

Figure 11.5.: Cluster within Figure 11.2: news pages and tools for hackers

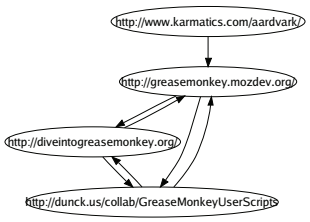

Figure 11.6.: Cluster within Figure 11.2: the GreaseMonkey extension for Firefox

12. Outlook

In the preceding chapters, we have discussed two paradigms for self-organized collaborative knowledge management. In this chapter, we will conclude the thesis by having a look at current trends both in P2PKM as well as in folksonomies and the Web 2.0 in general.

In this thesis, we have explored two paradigms for self-organized, collaborative knowledge management. In P2PKM, the goal is to collaboratively work with rich, semantic descriptions of resources, distributing them in the network and obtaining new annotations from others; on the other hand, folksonomies have been centralized efforts from the beginning, collecting their light-weight knowledge representation in a central server. For both approaches, we have offered solutions to some of their particular problems and challenges in the previous chapters.

The question remains, however, what the future trends and issues of P2PKM and folksonomies will be. In the remainder of this chapter, we will explore some recent work in both fields and point out possible future research directions.

12.1. Peer-to-Peer Knowledge Management

12.1.1. Combining Semantic P2PKM and DHTs

If one assesses the work done over the past years in P2P systems at events such as the IEEE P2P Conferences[1], and the IPTPS[2] and AP2PC[3] workshop series, most of the work regarding query routing has been done on so-called distributed hash tables (DHT). In DHTs, binary keys, e. g., hash values of file names, are mapped to network addresses of

[1] http://p2p2006.csc.ncsu.edu/
[2] http://iptps06.cs.ucsb.edu/
[3] http://p2p.ingce.unibo.it/2006/

peers that are responsible for storing the respective data. By building data structures such as trees (Aberer et al., 2003a) or skip lists (Stoica et al., 2001) on the key space, routing algorithms can be enabled which find data items with average costs that are logarithmic in the network size, both in terms of network load as well as routing table size.

While DHTs have provably small routing costs, the main drawback is that in their basic form, they support only the exact retrieval of binary keys, which makes them unsuitable for the P2PKM use cases sketched in this thesis. Benefits of the proposed self-organized network topologies such as peer autonomy, formation of topically related neighborhoods, similarity queries, or browsing of the surroundings of interesting query hits cannot easily be implemented in DHTs.

One possible research agenda for building semantic P2PKM solutions on top of DHTs would be as follows (Aberer, 2005):

1. Find a mapping of rich knowledge representation formalisms such as RDFS or OWL ontologies into an n-dimensional space that preserves the semantic distance of ontology entities.

2. If $n > 1$, find a mapping of that space into the set of binary keys permissible in a DHT.

3. Design a DHT that works well with a distance-preserving hash function and allows for range queries.

4. If peer autonomy is to be maintained, introduce a level of indirection between peers keeping the actual data and peers keeping the routing information about data objects.

Some partial solutions to some of these problems are already available, e. g., DHTs supporting range queries (Karnstedt et al., 2006) (step 3 above)—and some are rather easy to implement, such as step 4. To the best of the author's knowledge, however, no full solution to the problem of mapping rich knowledge representations to DHTs while preserving the all possible ways of accessing and using such knowledge bases has been found. Solutions proposed so far, e. g., (Cai and Frank, 2004; Aberer et al., 2004), are restricted to storing RDF triples at three locations each, indexed by their subjects, predicates, and objects, or to managing all facts pertaining to one particular concept of an ontology at one designated peer in a hypercube topology (Schlosser et al., 2002).

Implementing the abovementioned steps would be an interesting combination of two P2P paradigms and would open up additional possibil-

ities, e. g., integrating queries on the conceptual structure with queries for literal values.

12.1.2. Social Networks and P2PKM

While the analysis and study of social networks has a long tradition (Wasserman and Faust, 1994), only recently has the computer science and knowledge management community begun to investigate different uses of social networks (Staab et al., 2005), particularly in the P2PKM domain.

Here, social network metaphors have been used to structure network topologies, as proposed in Chapter 5, where network topologies are built around shared interests. Other aspects of social networks have been exploited as well, e. g., for evaluating the reputation and trust in peers (Wang et al., 2006).

Social networks in P2P systems could also be exploited to provide mixed modes of interaction with the network, e. g., by combining the querying of the network with browsing of the contents of friends' peers, where serendipitous discovery of useful contents is most likely.

12.2. Folksonomies

Folksonomies as a rather new phenomenon have seen considerable attention recently. While there has been a lot of discussion traditionally on Web 2.0 affine media such as blogs and mailing lists, there were relatively few peer-reviewed publications on folksonomies until recently.

While early published work was focused on classifying types of folksonomies, discussing their general properties, and relating them to other forms of knowledge organization, at the time of this writing research on folksonomies is diversifying into several more specialized directions. In the following, we will highlight some of the upcoming areas of research and point out possible future research paths.

Presentation: The user interface of the classic folksonomy tool was restricted to tag and user pages and a start page with recent posts. Recently, large folksonomy tools, most prominently del.icio.us, are beginning to restructure their user interfaces. Presumably this is intended to provide more interesting starting points for folksonomy exploration, and to raise the bar for spammers to get on the entry page. Future work will have to explore possible ways

of presenting large-scale folksonomies to their users; current work on user interface issues includes (Millen et al., 2006; Kaser and Lemire, 2007; Dubinko et al., 2006; Hassan-Montero and Herrero-Solana, 2006).

Ranking and Recommendations: Recent work (Niwa et al., 2006; Xu et al., 2006; Michail, 2005) as well as this thesis have proposed methods for ranking folksonomy contents and providing tag, user, and resource recommendations. In order to apply these methods in real-time to web-scale folksonomies (i. e., to rank query results at the time of user interaction), the performance would have to be improved drastically. Failing that, one has to resort to simpler ways of ranking and recommending or to precomputed results that are obtained in batch mode.

Providing Richer Structure: One area that has attracted the attention of many researchers from the data mining and Semantic Web communities is mining richer structure from folksonomies and provide it to the user without any added effort from the user's side. From early scientific papers on folksonomies, it has been attempted to extract structure from the flat space of tags. Open questions in this respect can be split into two classes: (a) *what* structures to provide, and (b) *how* to acquire or extract information about richer structures. So far, some attempts have been made to attack the latter problem and extract taxonomies or ontologies from folksonomies (Mika, 2005; Schmitz, 2006; Heymann and Garcia-Molina, 2006); the former question, namely, what kind of additional structure is most suitable in order to help folksonomy end users, is still open (Tonkin and Guy, 2006). Building richer structures from folksonomies or in a folksonomy-like environment is also a current topic of research in the Semantic Web community; see the next section for more details.

Folksonomy Dynamics and Trend Detection: Another interesting area of research is the detection of trends over time in large-scale folksonomy data (Hotho et al., 2006b; Dubinko et al., 2006), and of the dynamic processes in general that govern the growth of a folksonomy (Cattuto et al., 2007a; Shepard et al., 2006). We are currently exploring the connection between trends regarding the same topics in folksonomies and classic search engines, e. g., in order to see if folksonomies have a head start when trends are appearing on the web.

12.3. Web 2.0 and the Semantic Web

Probably the most dynamic development regarding the topics of this thesis is the convergence of ideas from the Web 2.0 area with established Semantic Web research. On the one hand, the Semantic Web community is interested in the impressive growth of Web 2.0 tools such as folksonomies, blogs, wikis, and similar sites within a very short time, and the impetus that collaborative management of knowledge generates—at least in many cases such as del.icio.us or Wikipedia. On the other hand, the ease of use, quick responsiveness, and large user base of Web 2.0 tools cannot replace richer knowledge representation formalisms in all cases, and many users want structure beyond fulltext blogs and wikis and a flat set of tags.

Unsurprisingly, many ways of combining the benefits of lightweight collaborative solutions in the Web 2.0 style and Semantic Web knowledge representations are being discussed today, and concrete projects are under way. In the remainder of this section, we will showcase some of the current work in the intersection of the Web 2.0 and the Semantic Web.

Folksonomies as Ontology Substrates. As has been discussed in Chapter 7, folksonomies are rather close to what can be considered simple ontologies on the scale of Smith and Welty (2001). Recently, several new approaches have appeared which regard folksonomies as transitional structures on the way to more elaborate knowledge representations (Braun et al., 2007), or which use already available background knowledge from Wikipedia[4] or WordNet (Fellbaum, 1998) to semantically enrich tags added by users (Marchetti et al., 2007).

Similar to this approach—though not based on folksonomies—is the idea of gleaning RDF metadata from HTML or XML documents (Davis, 2006), or to attaching semi-structured metadata to content using so-called *microformats* (Khare, 2006).

Collaborative Ontology Editing. Understanding ontologies as *shared* conceptualizations implies that there needs to be a way of eliciting these shared understandings of a domain from possibly dispersed sets of stakeholders. While ontology engineering methodologies have addressed this issue in the past (Sure, 2003; Tempich et al.,

[4] http://www.wikipedia.org

2006), the interest in collaboration on the Web 2.0 has produced several new tools.

As an example, *Soboleo* (Zacharias and Braun, 2007) is a tool for collaborative tagging that takes the tags from a taxonomy instead of a flat space. The taxonomy consists only of broader–narrower relationships and can be edited in the same web interface as the annotations, reducing the overhead of ontology construction.

On the other end of the spectrum, tools for editing classic Semantic Web ontologies expressed in RDFS or OWL have been extended to allow for collaborative editing of ontologies across the network, including facilities for discussing and voting on desired changes in the ontology (Tudorache and Noy, 2007; Kozaki et al., 2007).

Semantic Wikis. Although wikis have gained widespread use as one possible means of managing knowledge—Wikipedia being the most prominent example—they are still lacking an easy way of obtaining formal representations of their contents which can be reused, combined, and used for reasoning.

There are currently several projects which try to provide an environment for collaborative construction of structured knowledge representations in a wiki fashion, either editing the knowledge representation as such (Auer et al., 2007; Aumueller, 2005) or the structured knowledge along with the usual fulltext content (Schaffert, 2006; Krötzsch et al., 2006).

12.4. Conclusion

In the author's opinion, the solutions sketched in the last section are currently the subject of the most dynamic research within the field discussed in this thesis. Combining the ease of use and little overhead of Web 2.0 tools with the rich knowledge representations that are used in the Semantic Web community carries significant allure, although the Semantic Web community seems more interested in the work of the more grass-roots Web 2.0 movement than the other way around. One of the most interesting challenges will be to find the proper mixture to provide on the one hand enough formal semantics and sound reasoning capabilities for the respective application, while on the other hand keeping a KM system simple enough to use so that significant numbers of users will be able and willing to contribute.

Bibliography

A. Abecker. *Business-process oriented knowledge management: Concepts, methods, and tools*. PhD thesis, University of Karlsruhe, June 2004.

K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-Grid: a self-organizing structured P2P system. *ACM SIGMOD Record*, 32(3):29–33, 2003a.

K. Aberer. Personal communication. September 2005.

K. Aberer, P. Cudré-Mauroux, and M. Hauswirth. The chatty web: Emergent semantics through gossiping. In *Proc. 12th International World Wide Web Conference*, pages 197–206. Budapest, Hungary, May 2003b.

K. Aberer, P. Cudré-Mauroux, M. Hauswirth, and T. V. Pelt. GridVine: Building internet-scale semantic overlay networks. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *The Semantic Web—ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7–11, 2004. Proceedings*, volume 3298 of *Lecture Notes in Computer Science*, pages 107–121. Springer, Berlin, Heidelberg, 2004.

R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. SIGMOD Conf.*, pages 207–216, 1993.

B. Ahlborn, W. Nejdl, and W. Siberski. OAI-P2P: A peer-to-peer network for open archives. In *31st International Conference on Parallel Processing Workshops (ICPP 2002 Workshops), 20–23 August 2002, Vancouver, BC, Canada*, pages 462–468. IEEE Computer Society, 2002.

H. Alani, S. Dasmahapatra, K. O'Hara, and N. Shadbolt. Identifying communities of practice through ontology network analysis. *IEEE Intelligent Systems*, 18(2):18–25, March / April 2003.

G. Antoniou and F. van Harmelen. *A semantic web primer*. The MIT Press, Cambridge, MA, 2004.

S. Auer, S. Dietzold, J. Lehmann, and T. Riechert. OntoWiki: A tool for social, semantic collaboration. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

D. Aumueller. Towards a semantic wiki experience—desktop integration and interactivity in WikSAR. In *Proceedings of the ISWC 2005 Workshop on The Semantic Desktop*. Galway, Ireland, November 6 2005.

M. Aurnhammer, P. Hanappe, and S. L. Integrating collaborative tagging and emergent semantics for image retrieval. In *Proc. Collaborative Web Tagging Workshop, WWW 2006*. Edinburgh, Scotland, May 2006.

F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The description logic handbook: Theory, implementation and applications*. Cambridge University Press, Cambridge, 2003.

A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: Authority-based keyword search in databases. In M. A. Nascimento, M. T. Özsu, D. Kossmann, R. J. Miller, J. A. Blakeley, and K. B. Schiefer, editors, *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31–September 3, 2004*, pages 564–575. Morgan Kaufman, San Francisco, 2004.

A.-L. Barabási. *Linked: How everything is connected to everything else and what it means*. Plume, New York, 2003.

A. Barrat, M. Barthelemy, and A. Vespignani. Weighted evolving networks: coupling topology and weights dynamics. *Physical Review Letters*, 92:228701, 2004.

M. Barthelemy, A. Barrat, R. Pastor-Satorras, and A. Vespignani. Velocity and hierarchical spread of epidemic outbreaks in scale-free networks. *Physical Review Letters*, 92:178701, 2004.

K. Beck. *eXtreme programming—embrace change*. Addision-Wesley, Boston, 2000.

C. Berge. *Graphs and hypergraphs*. North Holland, Amsterdam, 1985.

C. Berge. *Hypergraphs*. North Holland, Amsterdam, 3rd edition, 1989.

T. Berners-Lee. *Weaving the web*. HarperSanFrancisco, San Francisco, 1999.

T. Berners-Lee, R. Fielding, and L. Masinter. RFC 3986, uniform resource identifier (URI): Generic syntax. 2005. http://rfc.net/rfc3986.html.

T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

S. Bloehdorn, O. Görlitz, S. Schenk, and M. Völkel. TagFS—tag semantics for hierarchical file systems. In *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06), Graz, Austria, September 6–8, 2006*, September 2006.

B. Bollobas. *Random graphs*. Cambridge University Press, Cambridge, New York, 2001.

M. Bonifacio, P. Bouquet, P. Busetta, A. Danieli, A. Donà, G. Mameli, and M. Nori. KEEx: A peer-to-peer solution for distributed knowledge management. In *Proceedings of the MobiQuitous Workshop on Peer-to-Peer Knowledge Management (P2PKM 2004)*. Boston, MA, USA, August 2004.

C. Borgelt. Recursion pruning for the apriori algorithm. In R. Bayardo Jr., B. Goethals, and M. J. Zaki, editors, *FIMI*, volume 126 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.

E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, and V. Zacharias. KAON—towards a large scale semantic web. In K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, editors, *EC-Web*, volume 2455 of *Lecture Notes in Computer Science*, pages 304–313. Springer, Berlin, Heidelberg, 2002.

J. Brase and W. Nejdl. Ontologies and metadata for eLearning. In R. Studer and S. Staab, editors, *Handbook on Ontologies in Information Systems*, pages 579–598. Springer Verlag, Berlin, Heidelberg, New York, 2003.

S. Braun, A. Schmitz, and A. Walter. Ontology maturing: A collaborative Web 2.0 approach to ontology engineering. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, editors. *Extensible markup language (xml) 1.0*. World Wide Web Consortium, 4th edition, 2006.

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, April 1998.

J. Broekstra, M. Ehrig, P. Haase, F. Harmelen, M. Menken, P. Mika, B. Schnizler, and R. Siebes. Bibster—a semantics-based bibliographic peer-to-peer system. In *Proc. WWW'04 Workshop on Semantics in Peer-to-Peer and Grid Computing*. New York, May 2004.

W. J. Brown, R. C. Malveau, H. W. M. III, and T. J. Mowbray. *Antipatterns: Refactoring software, architectures, and projects in crisis*. John Wiley & Sons, New York, 1998.

M. Cai and M. R. Frank. RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network. In S. I. Feldman, M. Uretsky, M. Najork, and C. E. Wills, editors, *Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004*, pages 650–657. ACM, 2004.

C. Cattuto, V. Loreto, and L. Pietronero. Semiotic dynamics and collaborative tagging. *Proceedings of the National Academy of Sciences United States of America*, 104:1461, 2007a.

C. Cattuto, C. Schmitz, A. Baldassarri, V. D. P. Servedio, V. Loreto, A. Hotho, M. Grahl, and G. Stumme. Network properties of folksonomies. *AI Communications Special Issue on "Network Analysis in Natural Sciences and Engineering" (to appear)*, 2007b.

S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11–16):1623–1640, 1999.

P.-A. Chirita, S. Costache, W. Nejdl, and R. Paiu. Beagle^{++}: Semantically enhanced searching and ranking on the desktop. In Sure and Domingue (2006), pages 348–362.

R. Cole and G. Stumme. CEM—a conceptual email manager. In B. Ganter and G. W. Mineau, editors, *Conceptual Structures: Logical, Linguistic, and Computational Issues*Proc. ICCS '00, volume 1867 of *LNAI*, pages 438–452. Springer, Heidelberg, 2000.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. The MIT Press, Cambridge, MA, 2nd edition, 2001.

A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proc. International Conference on Distributed Computing Systems (ICDCS)*, page 23. Vienna, Austria, July 2002.

T. Davenport and L. Prusak. *Working knowledge: How organizations manage what they know*. Harvard Business School Press, Boston, 1998.

T. Davenport. Information technologies for knowledge management. In Ichijo and Nonaka (2006), pages 97–117.

I. Davis. GRDDL primer. 2006. http://www.w3.org/TR/2006/WD-grddl-primer-20061002/.

J. de Bruijn, F. Martin-Recuerda, D. Manov, and M. Ehrig. State-of-the-art survey on ontology merging and aligning (SEKT project deliverable 4.2.1). http://sw.deri.org/~jos/sekt-d4.2.1-mediation-survey-final.pdf, 2004.

S. Decker and M. R. Frank. The networked semantic desktop. In *Proc. WWW Workshop on Application Design, Development and Implementation Issues in the Semantic Web*. New York, May 2004.

S. Decker, J. Park, D. Quan, and L. Sauermann, editors. *The semantic desktop—next generation information management & collaboration infrastructure. proc. of semantic desktop workshop at the iswc 2005*, volume 175 of *CEUR Workshop Proceedings ISSN 1613–0073*. November 2005.

S. Decker, J. Park, L. Sauermann, S. Auer, and S. Handschuh, editors. *Proceedings of the semantic desktop and social semantic collaboration workshop (semdesk 2006) at the iswc 2006*, volume 202 of *CEUR-WS*. November 2006.

P. Dourish, W. K. Edwards, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. B. Terry, and J. Thornton. Extending document management systems with user-specific active properties. *ACM Trans. Inf. Syst.*, 18(2):140–170, 2000.

M. Dubinko, R. Kumar, J. Magnani, J. Novak, P. Raghavan, and A. Tomkins. Visualizing tags over time. In L. Carr, D. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, editors, *Proceedings of the 15th international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, UK, May 23-26, 2006*, pages 193–202. ACM, 2006.

M. Ehrig, C. Tempich, J. Broekstra, F. van Harmelen, M. Sabou, R. Siebes, S. Staab, and H. Stuckenschmidt. SWAP—ontology-based knowledge management with peer-to-peer technology. In Y. Sure and H.-P. Schnurr, editors, *WOW2003, Workshop Ontologie-basiertes Wissensmanagement (German Workshop on Ontology-based Knowledge Management), Proceedings, Luzern, 2.-4. April, 2003*, volume 68 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2003.

European Commission. *Facing the challenge—the Lisbon strategy for growth and employment*. Office for Official Publications of the European Communities, Luxembourg, 2004.

C. D. Fellbaum. *WordNet—an electronic lexical database*. MIT Press, Cambridge, MA, 1998.

S. Ferré and O. Ridoux. Searching for objects and properties with logical concept analysis. In H. S. Delugach and G. Stumme, editors, *Conceptual Structures: Broadening the Base, 9th International Conference on Conceptual Structures, ICCS 2001, Stanford, CA, USA, July 30-August 3, 2001, Proceedings*, volume 2120 of *Lecture Notes in Computer Science*, pages 187–201. Springer, 2001.

W. B. Frakes and R. Baeza-Yates. *Information retrieval: Data structures & algorithms*. Prentice-Hall, Englewood Cliffs, NJ, 1992.

H. Frey, D. Görgen, J. K. Lehnert, and P. Sturm. A Java-based uniform workbench for simulating and executing distributed mobile applications. In *Scientific Engineering of Distributed Java Applications, Third International Workshop, FIDJI 2003*, pages 116–127. Luxembourg, November 2003.

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns—elements of reusable object-oriented software*. Addison-Wesley, Reading, MA, 1995.

B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical foundations*. Springer, Berlin, Heidelberg, 1999.

D. Gendarmi, F. Abbattista, and F. Lanubile. Fostering knowledge evolution through community-based participation. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

D. Gentner and S. K. Brem. Is snow really like a shovel? Distinguishing similarity from thematic relatedness. In M. Hahn and S. C. Stoness, editors, *Proc. Twenty-First Annual Meeting of the Cognitive Science Society*, pages 179–184. Mahwah, NJ, 1999.

Y. Goland, E. Whitehead, A. Faizi, S. Carter, and D. Jensen. HTTP extension for distributed authoring – WebDAV (RFC 2158). 1999. http://www.ietf.org/rfc/rfc2518.txt.

T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. In *Proc. International Workshop on Formal Ontology*. Padova, Italy, March 1993.

M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, H. F. Nielsen, A. Karmarkar, and Y. Lafon. SOAP version 1.2 part 1: Messaging framework. 2006. http://www.w3.org/TR/2006/PER-soap12-part1-20061219/.

P. Haase, R. Siebes, and F. van Harmelen. Peer selection in peer-to-peer networks with semantic topologies. In M. Bouzeghoub, C. A. Goble, V. Kashyap, and S. Spaccapietra, editors, *Semantics for Grid Databases, First International IFIP Conference on Semantics of a Networked World: ICSNW 2004, Paris, France, June 17-19, 2004. Revised Selected Papers*, volume 3226 of *Lecture Notes in Computer Science*, pages 108–125. Springer, Berlin, Heidelberg, 2004.

U. Hahn and U. Reimer. Knowledge-based text summarization: Salience and generalization operators for knowledge base abstraction. In Mani and Maybury (1999), pages 215–232.

H. Halpin. Identity, reference, and meaning on the web. In *Proc. WWW 2006 Workshop on Identity, Reference, and the Web*. Edinburgh, May 2006.

T. Hammond, T. Hannay, B. Lund, and J. Scott. Social bookmarking tools (I): A general review. *D-Lib Magazine*, 11, April 2005.

N. C. Hang and S. K. Cheung. Peer clustering and firework query model. In *Proc. 11th International World Wide Web Conference*. Honolulu, Hawaii, May 2002.

Y. Hassan-Montero and V. Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *Proc. InSciT 2006*. Merida, Spain, October 2006.

M. Hatala and G. Richards. POOL, POND and SPLASH: A canadian infrastructure for learning object repositories. In *Proceedings od the 5th IASTED Int. Conference on Computers and Advanced Technology in Education (CATE 2002)*, pages 54–59. Cancun, Mexico, 2002.

F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, editors. *Building expert systems*. Teknowledge Series in Knowledge Engineering. Addison-Wesley, Reading, MA, 1983.

M. Hepp. Possible ontologies: How reality constraints building relevant ontologies. *IEEE Internet Computing*, 11(1):90–96, January / February 2007.

A. Heydon and M. Najork. Mercator: A scalable, extensible web crawler. *World Wide Web*, 2(4):219–229, December 1999.

F. Heylighen. *The encyclopedia of life-support systems*, chapter The Science of Self-Organization and Adaptivity. EOLSS Publishers Co. Ltd., Oxford, 2001.

P. Heymann and H. Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical Report 2006–10, Computer Science Department, Stanford University, April 2006.

A. Hotho and G. Stumme. Conceptual clustering of text clusters. In G. Kokai and J. Z. (Eds.), editors, *Proc. Fachgruppentreffen Maschinelles Lernen (FGML 2002)*, pages 37–45. Hannover, 2002.

A. Hotho, A. Maedche, and S. Staab. Ontology-based text clustering. In *Proc. of the IJCAI Workshop "Text Learning: Beyond Supervision"*. Seattle, August 2001.

A. Hotho, S. Staab, and G. Stumme. Explaining text clustering results using semantic structures. In N. Lavrac, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *Knowledge Discovery in Databases: PKDD 2003, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22–26, 2003, Proceedings*, volume 2838 of *Lecture Notes in Computer Science*, pages 217–228. Springer, 2003.

A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In Sure and Domingue (2006), pages 411–426.

A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Trend detection in folksonomies. In Y. S. Avrithis, Y. Kompatsiaris, S. Staab, and N. E. O'Connor, editors, *Semantic Multimedia, First International Conference on Semantics and Digital Media Technologies, SAMT 2006, Athens, Greece, December 6–8, 2006, Proceedings*, volume 4306 of *Lecture Notes in Computer Science*, pages 56–70. Springer, Berlin, Heidelberg, 2006b.

E. Hovy and C.-Y. Lin. Automated text summarization in SUMMARIST. In Mani and Maybury (1999), pages 81–94.

Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.*, 2(3):283–304, 1998.

K. Ichijo and I. Nonaka, editors. *Knowledge creation and management: New challenges for managers*. Oxford University Press, New York, 2006.

I. Jacobson, G. Booch, and J. Rumbaugh. *The unified software development process*. Addison-Wesley, Reading, MA, 1999.

R. Jäschke, A. Hotho, C. Schmitz, B. Ganter, and G. Stumme. TRIAS—an algorithm for mining iceberg tri-lattices. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18–22 December 2006, Hong Kong, China*, pages 907–911. IEEE Computer Society, 2006.

G. Jeh and J. Widom. Scaling personalized web search. In *Proc. 12th International World Wide Web Conference*, pages 271–279. New York, NY, USA, 2003.

A. Jhingran. Enterprise information mashups: Integrating information, simply. In U. Dayal, K.-Y. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha, and Y.-K. Kim, editors, *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12–15, 2006*, pages 3–4. ACM, 2006.

S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Exploiting the block structure of the web for computing PageRank (preprint). 2003a. http://www.stanford.edu/~sdkamvar/papers/blockrank.pdf.

S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Extrapolation methods for accelerating PageRank computations. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 261–270. ACM Press, New York, NY, USA, 2003b.

M. Karnstedt, K.-U. Sattler, M. Hauswirth, and R. Schmidt. Cost-aware processing of similarity queries in structured overlays. In *Peer-to-Peer Computing*, pages 81–89. IEEE Computer Society, 2006.

O. Kaser and D. Lemire. Tag-cloud drawing: Algorithms for cloud visualization. In *Proc. WWW 2007 Workshop on Tagging and Metadata for Social Information Organization*. Banff, Canada, May 2007.

L. Kaufman and P. J. Rousseeuw. *Finding groups in data: An introduction to cluster analysis.* John Wiley, New York, 1990.

R. Khare. Microformats: The next (small) thing on the semantic web? *IEEE Internet Computing*, 10(1):68–75, 2006.

J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

D. E. Knuth. *The art of computer programming, volume II: Seminumerical algorithms.* Addison-Wesley, Reading, MA, 2nd edition, 1981.

K. Kozaki, E. Sunagawa, Y. Kitamura, and R. Mizoguchi. Distributed and collaborative construction of ontologies using Hozo. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

D. Krackhardt and J. Hanson. Informal networks: The company behind the chart. *Harvard Business Review*, 71(4):104–111, July / August 1993.

M. Krötzsch, D. Vrandecic, and M. Völkel. Semantic mediawiki. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *International Semantic Web Conference*, volume 4273 of *Lecture Notes in Computer Science*, pages 935–942. Springer, Berlin, Heidelberg, 2006.

F. Lehmann and R. Wille. A triadic approach to Formal Concept Analysis. In G. Ellis, R. Levinson, W. Rich, and J. F. Sowa, editors, *Conceptual Structures: Applications, Implementation and Theory*, volume 954 of *Lecture Notes in Computer Science*, pages 32–43. Springer, Berlin, Heidelberg, 1995.

P. G. Lind, M. C. Gonzalez, and H. J. Herrmann. Cycles and clustering in bipartite networks. *Phys. Rev. E*, 72(5), November 2005.

Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

A. Löser, C. Tempich, B. Quilitz, S. Staab, W. T. Balke, and W. Nejdl. Searching dynamic communities with personal indexes. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *Proc. 4th International Semantic Web Conference, ISWC 2005*, volume 3729 of *LNCS*, pages 491–505. Springer, Berlin, Heidelberg, November 2005.

C. Lucas. Self-organizing systems (SOS) FAQ. 2002. http://www.calresco.org/sos/sosfaq.htm.

A. Maedche and S. Staab. Measuring similarity between ontologies. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1–4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 251–263. Springer, 2002.

A. Maedche and R. Volz. The Text-To-Onto ontology extraction and maintenance system. In *Proc. ICDM 2001 Workshop on Integrating Data Mining and Knowledge Management*. San Jose, CA, USA, November 2001.

A. Maedche, M. Ehrig, S. Handschuh, R. Volz, and L. Stojanovic. Ontology-focused crawling of documents and relational metadata. In *Proceedings of the Eleventh International World Wide Web Conference WWW-2002*. Honolulu, Hawaii, May 2002a. (Poster).

A. Maedche, V. Pekar, and S. Staab. Ontology learning part one—on discovering taxonomic relations from the web. In N. Zhong, J. Liu, and Y. Yao, editors, *Web Intelligence*, pages 301–319. Springer, Berlin, Heidelberg, 2002b.

A. Maedche, B. Motik, and L. Stojanovic. Managing multiple and distributed ontologies on the Semantic Web. *VLDB Journal*, 12(4):286–302, November 2003.

R. Maier. *Knowledge management systems: Information and communication technologies for knowledge management*. Springer, Berlin, Heidelberg, 2nd edition, 2005.

I. Mani and M. T. Maybury, editors. *Advances in automatic text summarization*. MIT Press, Cambridge, MA, 1999.

A. Marchetti, M. Tesconi, F. Ronzano, M. Rosella, and S. Minutoli. SemKey: A semantic collaborative tagging system. In *Proc. WWW 2007 Workshop on Tagging and Metadata for Social Information Organization*. Banff, Canada, May 2007.

C. Marlow, M. Naaman, D. Boyd, and M. Davis. Position paper, tagging, taxonomy, flickr, article, toRead. In *Proc. Collaborative Web Tagging Workshop, WWW2006*. Edinburgh, 2006.

A. Mathes. Folksonomies—cooperative classification and communication through shared metadata. December 2004. http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html.

H. Maurer and M. Sapper. E-Learning has to be seen as part of general knowledge management. In *Proceedings of ED-MEDIA 2001*, pages 1249–1253. AACE, Charlottesville, USA, 2001.

S. Mazzocchi. Folksologies: de-idealizing ontologies. April 2005. http://www.betaversion.org/~stefano/linotype/news/85/.

D. L. McGuinness. Ontologies come of age. In D. Fensel, J. A. Hendler, H. Lieberman, and W. Wahlster, editors, *Spinning the Semantic Web*, pages 171–194. MIT Press, Cambridge, MA, 2003.

P. Merholz. Metadata for the masses. October 2004. http://adaptivepath.com/publications/essays/archives/000361.php.

P. Mika. Ontologies are us: A unified model of social networks and semantics. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *The Semantic Web—ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6–10, 2005, Proceedings*, volume 3729 of *Lecture Notes in Computer Science*, pages 522–536. Springer, Berlin, Heidelberg, 2005.

A. Michail. CollaborativeRank: Motivating people to give helpful and timely ranking suggestions. April 2005. http://collabrank.web.cse.unsw.edu.au/collabrank.pdf.

S. Milgram. The small world problem. *Psychology Today*, 67(1):61–67, 1967.

D. R. Millen, J. Feinberg, and B. Kerr. Dogear: Social bookmarking in the enterprise. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 111–120. ACM Press, New York, NY, USA, 2006.

J. Monaco. *How to read a film: The world of movies, media, multimedia:language, history, theory*. Oxford University Press, 3rd edition, 2000.

A. Naeve, M. Nilsson, M. Palmér, and F. Paulsson. Contributions to a public e-learning platform: infrastructure; architecture; frameworks; tools. *International Journal of Learning Technology*, 1(3):352–381, 2005.

S. Näher and O. Zlotowski. Design and implementation of efficient data types for static graphs. In R. H. Möhring and R. Raman, editors, *Algorithms—ESA 2002, 10th Annual European Symposium, Rome, Italy, September 17–21, 2002, Proceedings*, volume 2461 of *Lecture Notes in Computer Science*, pages 748–759. Springer, Berlin, Heidelberg, 2002.

W. Nejdl. Semantic web and peer-to-peer technologies for distributed learning repositories. In *Proceedings of the IFIP 17th World Computer Congress*, pages 33–50. Kluwer, B.V., Deventer, The Netherlands, 2002.

W. Nejdl, B. Wolf, C. Qu, S. Decker, A. Naeve, M. Nilsson, M. Palmér, and T. Risch. Edutella: A P2P networking infrastructure based on RDF. In *Proceedings of the 11th International World Wide Web Conference*, pages 604–615. Honolulu, Hawaii, May 2002.

W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst, and A. Löser. Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks. In *Proceedings of the 12th International World Wide Web Conference*, pages 536–543. Budapest, May 2003.

M. Newman, A.-L. Barabasi, and D. J. Watts, editors. *The structure and dynamics of networks*. Princeton University Press, Princeton, NJ, USA, 2006.

C. H. Ng and K. C. Sia. Peer clustering and firework query model. In *Poster Proceedings of the 11th International World Wide Web Conference*. Honolulu, Hawaii, May 2002.

M. Nilsson and M. Palmér. Conzilla—towards a concept browser. Technical Report CID-53, TRITA-NA-9911, Centre for user-oriented Information technology Design (CID), Department of Scientific Computing (TDB), Uppsala University, 1999.

M. Nilsson and W. Siberski. RDF query exchange language (QEL)—concepts, semantics and RDF syntax. June 2003. http://edutella.jxta.org/spec/qel.html.

M. Nilsson, M. Palmér, and J. Brase. The LOM RDF binding—principles and implementation. In *Proceedings of the third annual ARIADNE conference*. Leuven, Belgium, November 2003.

S. Niwa, T. Doi, and S. Honiden. Web page recommender system based on folksonomy mining. In *Proc. Third International Conference on Information Technology: New Generations (ITNG 2006)*, pages 388–393. Las Vegas, April 2006.

I. Nonaka and H. Takeuchi. *The knowledge creating company*. Oxford University Press, New York, 1995.

I. Nonaka, R. Toyama, and N. Konno. SECI, Ba and leadership: A unified model of dynamic knowledge creation. *Long Range Planning*, 33: 5–34, 2000.

I. Nonaka and T. Nishiguchi. *Knowledge emergence: Social, technical, and evolutionary dimensions of knowledge creation*. Oxford University Press, New York, 2001.

A. Oram, editor. *Peer-to-peer*. O'Reilly, Sebastopol, CA, 2001.

T. O'Reilly. What is web 2.0. Design patterns and business models for the next generation of software. September 2005. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html.

L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report SIDL-WP-1999-0120, Stanford Digital Library Technologies Project, 1998. http://dbpubs.stanford.edu:8090/pub/1999-66.

J. C. Paolillo and S. Penumarthy. The social structure of tagging internet video on del.icio.us. In *40th Hawaii International International Conference on Systems Science (HICSS-40 2007), CD-ROM / Abstracts Proceedings, 3-6 January 2007, Waikoloa, Big Island, HI, USA*, page 85. IEEE Computer Society, 2007.

N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Closed set based discovery of small covers for association rules. In *Actes des 15èmes journées Bases de Données Avancées (BDA'99)*, pages 361–381. Bordeaux, October 1999.

A. Pothen. Graph partitioning algorithms with applications to scientific computing. In D. E. Keyes, A. Sameh, and V. Venkatakrishnan, editors, *Parallel Numerical Algorithms*, pages 323–368. Kluwer, Dordrecht, Boston, 1997.

G. Probst, S. Raub, and K. Romhardt. *Wissen managen: Wie Unternehmen ihre wertvollste Ressource optimal nutzen*. Gabler, Wiesbaden, 4th edition, 2003.

L. Prusak and L. Weiss. Knowledge in organizational settings: How organizations generate, disseminate, and use knowledge for their competitive advantage. In Ichijo and Nonaka (2006), pages 32–43.

D. Quan and D. R. Karger. How to make a semantic web browser. In *Proceedings of the 13th International World Wide Web Conference*, pages 255–265. New York City, NY, USA, May 2004.

R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 19(1):17–30, January / February 1989.

E. S. Raymond. The cathedral and the bazaar. *First Monday*, 3(3), 1998.

P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95*, pages 448–453. Montreal, Canada, August 1995.

P. Reynolds and A. Vahdat. Efficient peer-to-peer keyword searching. In M. Endler and D. C. Schmidt, editors, *Middleware 2003, ACM/IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, June 16–20, 2003, Proceedings*, volume 2672 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2003.

RSS Board. RSS 2.0 specification. 2006. http://www.rssboard.org/rss-specification.

L. Sauermann, G. A. Grimnes, M. Kiesel, C. Fluit, H. Maus, D. Heim, D. Nadeem, B. Horak, and A. Dengel. Semantic desktop 2.0: The Gnowsis experience. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web—ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5–9, 2006, Proceedings*, volume 4273 of *Lecture Notes in Computer Science*, pages 887–900. Springer, Berlin, Heidelberg, 2006.

N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. *Signal Processing Magazine, IEEE*, 23(2): 133–141, 2006.

S. Schaffert. IkeWiki: A semantic wiki for collaborative knowledge management. In *15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2006), 26–28 June 2006, Manchester, United Kingdom*, pages 388–396. IEEE Computer Society, 2006.

T. Schank and D. Wagner. Approximating clustering coefficient and transitivity. *Journal of Graph Algorithms and Applications*, 9(2):265–275, 2005.

T. C. Schelling. *Micromotives and macrobehavior*. W. W. Norton, New York, 1978.

M. T. Schlosser, M. Sintek, S. Decker, and W. Nejdl. HyperCuP—hypercubes, ontologies, and efficient search on peer-to-peer networks. In G. Moro and M. Koubarakis, editors, *Agents and Peer-to-Peer Computing, First International Workshop, AP2PC 2002, Bologna, Italy, July, 2002, Revised and Invited Papers*, volume 2530 of *Lecture Notes in Computer Science*, pages 112–124. Springer, Berlin, Heidelberg, 2002.

C. Schmitz. Untersuchung der Graphstruktur von Web-Communities am Beispiel der Informatik. Master's thesis, Universität Trier, October 2001.

C. Schmitz. Self-organization of a small world by topic. In *Proc. 1st International Workshop on Peer-to-Peer Knowledge Management*. Boston, MA, August 2004.

C. Schmitz, S. Staab, R. Studer, G. Stumme, and J. Tane. Accessing distributed learning Repositories through a Courseware Watchdog. In *Proc. E-Learn 2002: World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education*, pages 909–915. Montreal, October 2002.

C. Schmitz, A. Hotho, R. Jäschke, and G. Stumme. Content aggregation on knowledge bases using graph clustering. In Sure and Domingue (2006), pages 530–544.

C. Schmitz, A. Hotho, R. Jäschke, and G. Stumme. Mining association rules in folksonomies. In V. Batagelj, H.-H. Bock, A. Feligioj, and

A. Žiberna, editors, *Data Science and Classification—Proc. 10th Jubilee Conference of the International Foundation of Classification Societies*, pages 261–270. Springer, Berlin, Heidelberg, 2006b.

C. Schmitz, M. Grahl, A. Hotho, G. Stumme, C. Catutto, A. Baldassarri, V. Loreto, and V. D. P. Servedio. Network properties of folksonomies. In *Proc. WWW2007 Workshop on Tagging and Metadata for Social Information Organization*. Banff, May 2007.

P. Schmitz. Inducing ontology from Flickr tags. In *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*, May 2006.

G. Schreiber and R. de Hoog. *Knowledge engineering and management: The CommonKADS methodology*. MIT Press, Cambridge, MA, 1999.

H. Shepard, H. Halpin, and V. Robu. The dynamics and semantics of collaborative tagging. In *Proc. of the 1st Semantic Authoring and Annotation Workshop (SAAW 2006) at ISWC 2006*. Athens, GA, November 2006.

C. Shirky. Listening to Napster. In Oram (2001), pages 21–37.

C. Shirky. Ontology is overrated: Categories, links, and tags. 2005. http://www.shirky.com/writings/ontology_overrated.html.

B. Simon, D. Massart, F. van Assche, and E. Duval. Simple query interface specification. 2004. http://nm.wu-wien.ac.at/e-learning/inter/sqi/sqi.pdf.

A. Singh and M. Haahr. Topology adaptation in P2P networks using schelling's model. In *Proceedings of the First Workshop on Games and Emergent Behaviours in Distributed Computing Environments, colocated with PPSN-VIII*. Birmingham, UK, September 2004.

A. Singh and M. Haahr. Creating an adaptive network of hubs using Schelling's model. *Communications of the ACM*, 49(3):69–73, 2006.

B. Smith and C. Welty. Ontology: Towards a new synthesis. In C. Welty and B. Smith, editors, *Proceedings of the Second International Conference on Formal Ontology in Information Systems*, pages iii–x. Ongunquit, Maine, 2001.

S. Staab, H.-P. Schnurr, R. Studer, and Y. Sure. Knowledge processes and ontologies. *IEEE Intelligent Systems*, 16(1):26–34, 2001.

S. Staab, S. Santini, F. Nack, L. Steels, and A. Maedche. Emergent semantics. *IEEE Intelligent Systems,* 17(1):78–86, 2002.

S. Staab, P. Domingos, P. Mika, J. Golbeck, L. Ding, T. W. Finin, A. Joshi, A. Nowak, and R. R. Vallacher. Social networks applied. *IEEE Intelligent Systems*, 20(1):80–93, 2005.

L. Steels. The origins of ontologies and communication conventions in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 1 (2):169–194, October 1998.

I. Stoica, R. Morris, D. R. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. SIGCOMM*, pages 149–160. San Diego, August 2001.

L. Stojanovic. *Methods and tools for ontology evolution*. PhD thesis, University of Karlsruhe (TH), Germany, August 2004.

L. Stojanovic, S. Staab, and R. Studer. Knowledge technologies for the semantic web. In W. A. Lawrence-Fowler and J. Hasebrook, editors, *Proceedings of WebNet 2001—World Conference on the WWW and Internet, Orlando, Florida, October 23–27, 2001,* pages 1174–1183. AACE, 2001.

H. Stuckenschmidt, F. van Harmelen, W. Siberski, and S. Staab. Peer-to-peer and semantic web. In S. Staab and H. Stuckenschmidt, editors, *Semantic Web and Peer to Peer*. Springer, Berlin, Heidelberg, 2006.

G. Stumme. Efficient data mining based on formal concept analysis. In A. Hameurlain, R. Cicchetti, and R. Traunmüller, editors, *Proc. DEXA 2002*, volume 2453 of *LNCS*, pages 534–546. Springer, Berlin, Heidelberg, 2002a.

G. Stumme. Conceptual knowledge discovery with frequent concept lattices. FB4-Preprint 2043, TU Darmstadt, 1999.

G. Stumme. Using ontologies and formal concept analysis for organizing business knowledge. In J. Becker and R. Knackstedt, editors, *Wissensmanagement mit Referenzmodellen – Konzepte für die Anwendungssystem- und Organisationsgestaltung*, pages 163–174. Physica, Heidelberg, 2002b.

G. Stumme. A finite state model for on-line analytical processing in triadic contexts. In B. Ganter and R. Godin, editors, *ICFCA*, volume 3403 of *Lecture Notes in Computer Science,* pages 315–328. Springer, Berlin, Heidelberg, 2005.

G. Stumme and R. Wille, editors. *Begriffliche Wissensverarbeitung – Methoden und Anwendungen*. Springer, Berlin, Heidelberg, 2000.

Y. Sure. *Methodology, tools and case studies for ontology based knowledge management*. PhD thesis, University of Karlsruhe, May 2003.

Y. Sure and J. Domingue, editors. *The semantic web: Research and applications, 3rd european semantic web conference, eswc 2006, budva, montenegro, june 11–14, 2006, proceedings*, volume 4011 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, 2006.

B. Szekely and E. Torres. Ranking bookmarks and bistros: Intelligent community and folksonomy development. May 2005. http://torrez.us/archives/2005/07/13/tagrank.pdf.

J. Tane. *Query-based multicontexts for knowledge base browsing*. PhD thesis, University of Karlsruhe, Karlsruhe, January 2007.

J. Tane, C. Schmitz, and G. Stumme. Semantic resource management for the web: An elearning application. In *Proc. 13th International World Wide Web Conference (WWW 2004) (Alternate Track)*, pages 1–10. New York, May 2004.

C. Tempich, S. Staab, and A. Wranik. REMINDIN': Semantic query routing in peer-to-peer networks based on social metaphors. In W3C, editor, *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, pages 640–649. ACM, New York, USA, May 2004.

C. Tempich, H. S. Pinto, and S. Staab. Ontology engineering revisited: An iterative case study. In Sure and Domingue (2006), pages 110–124.

M. Tepper. The rise of social software. *netWorker*, 7(3):18–23, 2003.

E. Tonkin. Folksonomies: The fall and rise of plain-text tagging. *Ariadne*, (47), April 2006.

E. Tonkin and M. Guy. Folksonomies: Tidying up tags? *D-Lib*, 12(1), January 2006.

E. Tsui. Technologies for personal and peer-to-peer knowledge management. Technical report, CSC Leading Edge Forum, July 2002. http://www.csc.com/aboutus/lef/mds67_off/uploads/P2P_KM.pdf.

T. Tudorache and N. Noy. Collaborative Protégé. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

G. Tummarello, C. Morbidoni, and M. Nucci. Enabling semantic web communities with DBin: An overview. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *International Semantic Web Conference*, volume 4273 of *Lecture Notes in Computer Science*, pages 943–950. Springer, Berlin, Heidelberg, 2006.

A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

T. Vander Wal. Folksonomy. 2007. http://vanderwal.net/folksonomy.html.

T. Vander Wal. Explaining and showing broad and narrow folksonomies. February 2005. http://www.personalinfocloud.com/2005/02/explaining_and_.html.

G. von Krogh, K. Ichijo, and I. Nonaka. *Enabling knowledge creation: How to unlock the mystery of tacit knowledge and release the power of innovation*. Oxford University Press, New York, 2000.

C. Wagner. Collaborative knowledge management: Breaking the knowledge acquisition bottleneck. In *Proc. Information Resources Management Association International Conference*. New Orleans, May 2004.

W. Wang, L. Zhao, and R. Yuan. Improving cooperation in peer-to-peer systems using social networks. In *Proc. 20th International Parallel and Distributed Processing Symposium (IPDPS2006)*. Rhodes, Greece, April 2006.

S. Wasserman and K. Faust. *Social network analysis: Methods and applications*. Cambridge University Press, Cambridge, 1994.

D. J. Watts. *Small worlds—the dynamics of networks between order and randomneess*. Princeton University Press, Princeton, New Jersey, 1999.

D. J. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

C. A. Welty and D. A. Ferrucci. What's in an instance? Technical Report #94–18, Computer Science Dept., Rensselaer Polytechnic Institute, 1994.

R. Wille. Restructuring lattices theory: An approach based on hierarchies of concepts. In I. Rival, editor, *Ordered Sets*, pages 445–470. Reidel, Dordrecht, Boston, 1982.

World Wide Web Consortium. XML schema. 2004. http://www.w3.org/XML/Schema.

W. Xi, B. Zhang, Y. Lu, Z. Chen, S. Yan, H. Zeng, W. Ma, and E. Fox. Link fusion: A unified link analysis framework for multi-type interrelated data objects. In *Proc. 13th Internation World Wide Web Conference*, pages 319–327. New York, May 2004.

Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the Semantic Web: Collaborative tag suggestions. In *Proc. WWW 2006 workshop on collaborative Web tagging*. Edinburgh, Scotland, May 2006.

V. Zacharias and S. Braun. SOBOLEO—social bookmarking and lightweight engineering of ontologies. In *Proc. WWW 2007 Workshop on Social and Collaborative Construction of Structured Knowledge*. Banff, Canada, May 2007.

M. J. Zaki and C.-J. Hsiao. ChARM: An efficient algorithm for closed association rule mining. Technical Report #99–10, Computer Science Dept., Rensselaer Polytechnic Institute, October 1999.

The web references have been checked for reachability on April 19, 2007.