Why Google Isn't the Future. Really Not.

Robert Eberhardt, Mario M. Kubek, and Herwig Unger

Chair of Communication Networks, FernUniversität in Hagen, Germany

Abstract: Searching the Internet means today to use Google or any other big, centralised search engine. Besides wasting a huge amount of energy and other resources, those systems are neither trustworthy and scalable nor can they generate satisfying search results. After an analysis of the reasons for it, a new concept for a decentralised search engine is presented and some estimations on the complexity of search with it are given. The concept shall not be understood as the *ultima ratio*, but as a start of a discussion to find future search solutions.

1 Motivation

Asking Internet users about the performance of Google, today's biggest search engine, most people will give comments like 'great', 'amazing' or 'incredible'. Only a few will mention that it is sometimes quite difficult to describe their information needs with a few keywords (in particular new ones) or find in the plethora of results returned the really desired and useful ones. Even less will complain about a missing trust estimation, a lack of consideration of the user's context dependencies or disambiguation problems. And only some people will worry about the imperial power of Google, which even does not allow a verification of the recall or precision of results presented. Last but not least, it remains unclear, what Google may derive just from the content of webpages collected as well as from the origin and keywords of all the requests made. In general, besides the famous *PageRank*-Algorithm of Page and Brin [1], not too much important details are known about the inner functionality of Google.

While discussions on the commercial influence of Google's result rankings constantly appear in the media, in this article, mostly technical issues shall be in the foreground of the considerations. In [2], the main structure of today's search engines is given, which mostly consists of any (distributed) crawling system, an indexing mechanism, an of course huge data base and a respective web interface. In general, any big search engine is nothing else than a well sorted (indexed) copy of the World Wide Web (WWW).

However, this architecture has many disadvantages: On the one hand, in order to be able to present recent results, the crawler must frequently download any web pages. Therefore, to achieve a high coverage and actuality (web results should cover content that has been updated in the last 24 hours with a probability of at least 80 percent), would overload any today's available network capacities. Problems get bigger, once the hidden web (deep web) is considered besides regularly accessible HTML pages, too.

On the other hand, modern web topology models (like the evolving web graph model [3]) emanate from the fact that there are linear as well as exponential growth components, if the overall number of websites is considered. From the point of view of complexity theory, it becomes clear that there might be no memory space available to keep and archive all those sites in the future.

Summarising, it can be figured out that search in the tomorrow's Internet

1. must be carried out without establishing a copy of the entire web,
2. shall return results without any commercial or other third party influences or censorship,
3. ensures that the returned results are 100 percent recent.
4. is personalised using a search history without giving intimate or personal user details to any centralised authority,
5. and is a process to which every user must contribute with his/her resources and cooperation.

If 'the network is the computer' (McNeill, SUN, 1991), which 'shall learn from examples and the behaviour of the user to restructure itself ...' (von der Malsburg, Ruhr-University Bochum, 2001), this goal can be achieved. The following sections will present a concept to do so.

2 Preliminary Works

First tries to avoid centralised instances and to build a fully decentralised Internet service system dated back to 1997 with the introduction of the Web Operating System ($WOS^{(TM)}$)[4]. It realised a remote service execution with a search service using local warehouses containing a set of privately known neighbours of every node (and therefore forming a connected service network) and cooperative services of any other network member.
Later, around 1999, this idea has been made perfect in the peer-to-peer (P2P)

paradigm, mostly used within file-sharing systems [5] like *Gnutella* or *FreeNet* [6].

Jordan RITTER[1][7] and others usually argued that those system cannot scale, due to an exponentially growing number of messages generated by broadcasts used to support system connectivity and search. In fact, this claim is wrong.
Messages in P2P-systems are bounded by two mechanisms: a time-to-live (TTL) or a hop-counter determining the maximum number of forwards for any message (usually set to 7) and the rule that every message (identified by an own Message-ID) is only forwarded once by every node (no matter how often it has been received from different neighbours). Consequently, in a complete graph, the number of messages cannot exceed N^2, if N denotes the number of nodes. In technical systems like *Gnutella*, every node usually keeps only a limited number of connections open, e.g. 4. In this case, the maximal number of messages is limited by $4 \cdot N$ even.
While the number of messages linearly depends on the size of the network, the average distance of a desired information from the requesting node is usually lower than $O(ln(N))$, as it is known from the works on small world graphs initiated by Stanley MILGRAM in 1969 in [8]. In such a manner, P2P systems could become an excellent foundation of a new generation of Internet search systems.

Structured P2P-Systems like CHORD, PASTRY or CAN (just to cite a few, for an overview see [9]) made the search in decentralised systems more efficient by limiting the average expenditure of a search to $O(ln(N))$ but generated some overhead to maintain those structures and do not support multi-keyword searches per se. However, for the first time, the importance of specialised structures and their self-organisation/emergence have been underlined by those systems. To this approach, other works contribute as well that e.g. find clusters of nodes [10] and optimise their neighbourhood depending on content aspects [11, 12].

Meanwhile, most search engines upgraded their functionality by an introduction of user accounts and statistical evaluations of frequently used search phrases. In such a manner, search engines offer their users additional query terms to refine their search based on the keywords previously entered by many users along with the initial query [13], geographic location or profile information. By using these services, the users often provide sensitive personal details related to private life and the search engine can collect and store a bigger set

[1].. accidently having a leading position in the competing NAPSTER company from 1999-2000.

of information of the respective user and learns details about his or her special behavior and interests.

3 Processing Local Documents and Data

As an answer, in [14], a fully locally working agent for an improved web search has been introduced (see Fig. 1), which avoids the mentioned disadvantages and increases the security of personal user information. Therefore, the unrivalled huge and well-indexed databases as well as access mechanisms of the big search engines are combined with a local pre-processing agent. This agent has access to the local (and maybe confidential) files of the user and may even establish a fine-granular user profile. Since these data are kept local, there is no danger for privacy and security of them. After processing the current search words of the user and/or the knowledge the agent obtains from these local files and previous searches, a keyword suggestion or extension is presented to the user, which can be used or not to be sent as a query to the (remote) search engine, which then returns its results in the known manner.

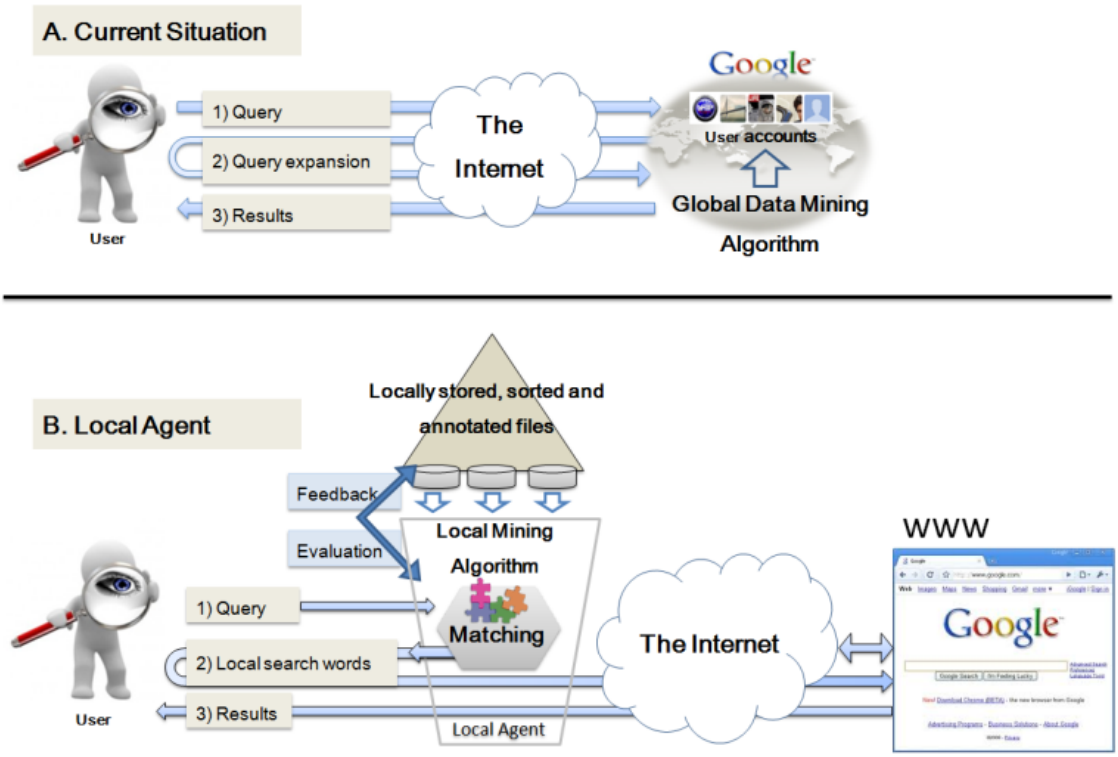

Fig. 1: Search with a local search agent

Later works of *KUBEK* extend the approach by the following refinements:

- full documents can be used as search queries and similar documents are returned;
- returned documents and links may be evaluated as positive or negative in order to refine the initial seed of given documents (FXResearcher, see [15]);
- local documents and those found in former searches may be analysed to derive more and more significant keywords for further search iterations (DocAnalyser, see [16]);
- search and download processes can be carried out continuously in the background, even if the user is offline;
- a selected search context may be used to analyse and evaluate links in a recently visited website to label them in respect to their importance for a visit [17];
- support of word-sense disambiguation by textual contexts of pre-selected pictures in web pages (PDSearch, see [18]);
- co-occurrences and associations (see [19]) may be used to distinguish more generalising or more specific keywords.

First approaches to P2P search engines are available, too. *YaCy* [20] and *FAROO* [21] are the most famous examples. Meta-searches [22, 23], the use of previous searches and results (as well as caching) [24] and approaches to collaborative filtering and search [25] complete the set of already realised approaches to improve the search in the Internet. However, all those approaches still adhere to the main working principle of a search engine and just re-organise crawling and indexing in a new, distributed manner using a higher number of peers.

Last but not least, in order to improve search processes, some efforts have been put into investigations in the area of ontologies [26], content annotations [27], text mining with text clustering [28], probabilistic methods [29], deep semantic analysis [30] as well as brain-like processing [31].

They all have influenced the work of the authors and shall therefore be mentioned here. Differing from the approaches cited above, the autors intend to build and maintain content and context depending structures in the network such that paths between queries and suitable documents can be obtained for any search processes.

4 Concept of a P2P-Web

4.1 The WebEngine

In the doctrine of most teachers and the knowledge of the users, the today's WWW is a classical client/server system. Web servers offer content to view or download using the HTTP protocol while every web browser is the respective client accessing content from any server. Clicking on a hyperlink in a web content usually means to change the used server, whose address is given in the URL of the link.

Nevertheless, any web server may also be regarded as a peer, which is connected to and therefore known by other peers through the addresses stored in the links of the hosted web pages. In such a manner, the WWW can be regarded as a P2P-system (with quite slow dynamics related to the addition or removal of peers).

In the suggested new concept for searching and using the WWW, especially the characteristics of web servers (or better called WebEngines) shall be significantly extended with P2P functionalities as it is to be seen in Fig. 2.

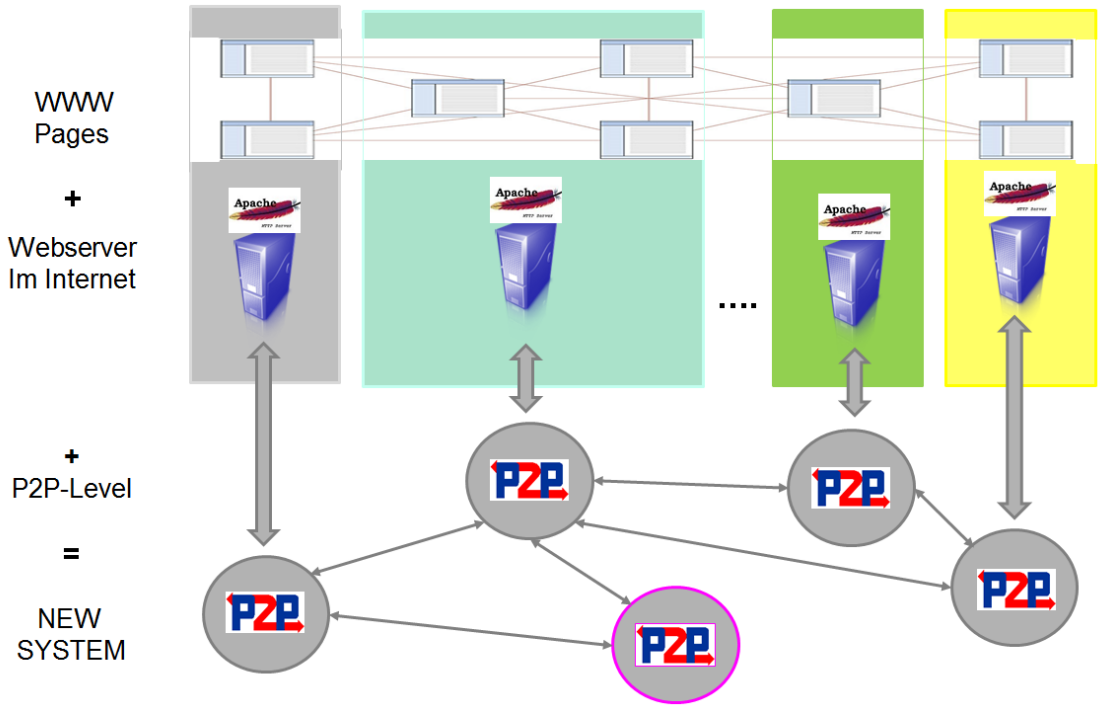

Fig. 2: First concept of a decentralised World Wide Web

Therefore, a P2P plug-in with a graphical user interface (GUI) for any standard web browser has been developed. It is connected to the frequently used Apache HTTP web server and may access the offered web pages and databases of the server with all related meta-information. In the first step, a classical, but multi-keyword P2P System is realised.

Therefore, the necessary functionality of the plug-in is the following:

1. The plug-in becomes an integral part of the web server as well as of the web browsers on the user/client-side. We later will call this unit a web peer.

2. A connected P2P-system shall be set up. Initially, the links contained in the bookmarks as well as the hosted web pages of the Apache HTTP server shall be used for this purpose. Other boot mechanism as known from [4] may be applied, too. Later, links may be added using the *PING/PONG*-protocol known from *Gnutella* and other P2P Systems. Note, that

   - nodes that do not yet run the P2P plug-in may be requested to install this software (e.g. nodes with a sufficient connectivity, performance and availability).

   - HTTP is used as frame protocol for any communication between the units.

   - A fixed number of connections will be kept open (although more possible neighbours are locally stored). Furthermore, the mechanisms mentioned above to limit the number of routed messages will be applied.

3. All hosted web documents will be indexed in a separate index file after applied stopword removal and stemming. The index is updated after every change in one of the hosted web documents.

4. The plug-in is able to generate a graphical interface, in particular a suitable search page for the requesting user. In the first version, the search is performed in correspondence with the *QUERY/HIT*-protocol of *Gnutella* (within HTTP frames). The only exception is that not substrings in file names are searched but search results will be generated through a search in the generated index files[2]. In such a manner, multi-keyword search will be possible.

[2]In the first versions, indexing will be limited to nouns as the carriers of meanings

5. A proliferation mechanism in the plug-in ensures the distribution of the system over the entire WWW. It shall be able to recognise the peer software on other web server addressed and offer the download of its own programme, in case the peer is not running at the destination yet.

The above specified system may rapidly change the kind of access, use and search in the WWW. As discussed, it may slowly grow besides the current WWW structures and make use of centralised search engines but may make them more and more obsolete. In this manner, the manipulation of search results through commercial influences as well as (the risks of) espionage will be greatly reduced.

Nevertheless, for this purpose, it is necessary to reach a critical mass of peers partaking in this P2P-web. In order to reach this goal, the P2P plug-in and its comprehensive installation and configuration instructions will be offered on a public website, from which users and especially administrators can directly download it. This website will offer an online service to instantly test the P2P system without having to install any software. too. Also, the user interface of the P2P plug-in will provide links to the software and social networks with options to share it. This way, any interested user coming across a web server running the P2P software will be able to get to know more about it and will hopefully recommend it. Additionally, the peer search functionalities will be accessible using search fields in web pages such as weblogs. This way, a seemless integration of the P2P functionality is reached and users can instantly benefit from its services by just making use of the query input fields in the usual manner. As more and more people will recognise the mentioned new search functionalities on many different web sites, they will be made aware of the software realising them, too.

Another way to distribute the P2P plug-in is getting in contact with web administrators not utilising the software yet. This can be achieved by monitoring the stream of visitors coming from websites. The P2P software can e.g. analyse the HTTP headers of incoming requests stored in log files. This way, it can find out using the Referer-field the address of the previously visited web page that links to the current server. It can then automatically issue standard P2P protocol messages to the originating server in order to test if it is running the P2P software already. If so and if there is a significant amount of users coming from this server, it should store this server's address in the list of peer neighbours as this constant stream of users indicates a valid interest in both web server's contents. If the originating server is not yet running the P2P software, the current web

server's P2P plug-in can try to identify the respective web administrator using DNS WHOIS lookups or by analysing the imprint of the remote website in order to determine the name, e-mail address and post address of its responsible webmaster using techniques known from natural language processing such as named-entity recognition. Once identified, the plug-in can suggest the current webserver's administrator to get in contact with the other webmaster to introduce the P2P software; for this purpose, the plug-in can conveniently provide an already filled contact form addressed to this person. The same procedure can of course be applied for web servers that are frequently linked to by the current web server, too. Many outgoing links to a destination express its relevance to the local content. Therefore and also in this case, it actually makes sense to contact the respective webmasters. Using these and further mechanisms, the P2P plug-in can be distributed with little effort and the P2P network will be established in a sustainable way by this means.

4.2 Structures

This $P2P - WWW$ system already realises the concept of a fully decentralised WWW. However, there are still significant possibilities to increase its performance and efficiency.

- At the beginning, not all documents from the current WWW are available in the P2P-system. However, once started with a small number of documents, the system can grow fastly as the P2P plug-in is installed in more web servers. Also, this growth shall also result in an increased chance for parallel computing.

- Document addition, removal and computing are a fully decentralised processes. As a result and in particular, no initial centroids or an initial number of clusters to obtain can be determined. Also, as already described in [31] the sequence of document addition or presentation may significantly influence the final results.

- A best fit to the one or the other group may be hard to define. Distance measures in the classic manner are hard to apply and cannot be considered as a means to derive fixed borders in the clustering/merging process. Especially, suitable cluster sizes and distances may significantly and strongly depend on the number of documents. I.e. small libraries may require less categorisation than bigger ones.

- The process is highly dynamic, i.e. already built clusters, assignments etc. may be waved fully or partially at any time by the one or the other participating unit.

These requirements attack the classic approach that are still present in other P2P search engines still giving web peers their own (distributed) index files. Similarities in the hosted web documents as well as shared groups of keywords do not automatically result in a grouping and joint management of those data items.

In the described approach, still every web peer is included in any search, what may generate a significant load especially for smaller machines.

The idea is to overcome this problem by a self-organising structure, where to a given search request a suitable path between the nodes representing the tokens of the request and the matching doument(s) can be found. Overlapping, tree-like (and no longer only locally hosted) index structures may help to overcome the isolation of index files on the web peers, as shown in Fig. 3.

Therefore, every single web document induces an undirected tree structure with

- the document identifier as the root node,
- as much leaves as needed to have exactly one for every word $w \in W_i$ in the document i (after stemming and stopword removal, words appearing several times in the document are regarded as types and therefore counted one time only) and
- $ln(|W|)$ intermediate tree levels that cover phrases or semantic relationships between the respective terms.

In fact, this approach is inspired and related to human cognitive processes involved in categorising literature sources.

As shown in Fig. 3b, the consideration of a group of documents results in identifying similar words and structures (subtrees) in those documents and builds -maybe (partially) overlapping- groups.

In order to establish bigger and fast growing libraries as well as to order them in a suitable distributed manner, appropriate methods for doing so must be conceived, developed and tested. Unfortunately, those words, patterns and structures are unknown in the beginning of the process, a circumstance that may

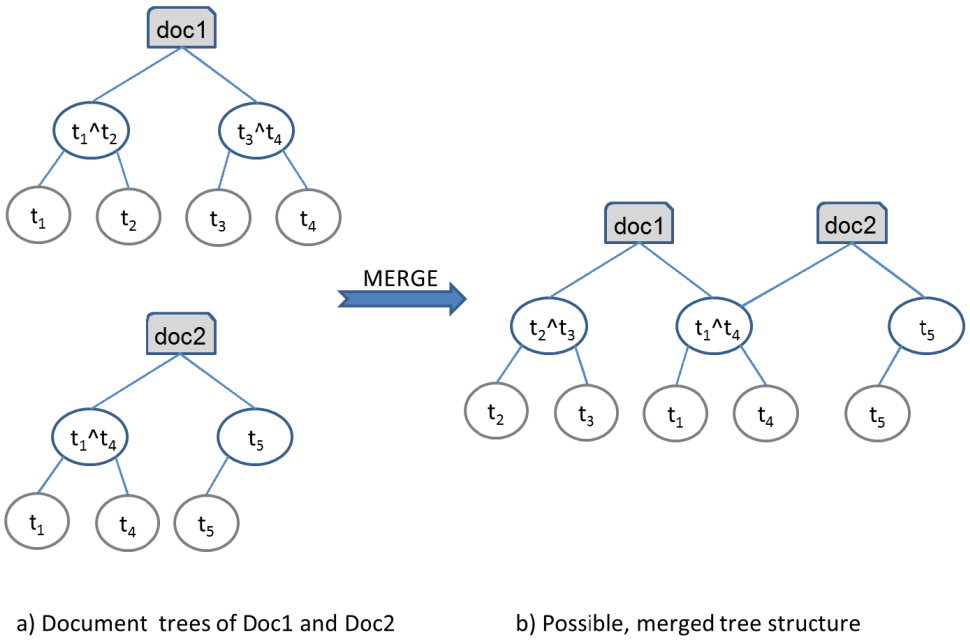

Fig. 3: Index trees of web documents

result in a later re-structering and destruction of already well-established and probably large structures. Also, it is must be accepted that the result is not uniquely determined, i.e. the same search conducted by different people under different circumstances and side-conditions may result in different structures and results, fulfilling only a few, generally accepted rules.

The respective concepts and processes for such a system are not fully conceived yet and will therefore be considered in further works.

5 Outlook

After analysing recent structures and operating paradigms of the WWW, a new architectural concept has been introduced. It is based on the extension of the already existing P2P-characteristics in the form of so called web peers and mechanisms for new, content-based structure building over the net. Hereby, a navigateable system of paths between tokens (words) and documents is built. The suggested system works in parallel to the existing web and can therefore be introduced and spread as a plug-in for web servers in the current WWW.
Significant research effort, however, is needed to use all opportunities to reduce traffic by the means of self-organisation, emergence, and structure building. First ideas exist and will be presented in future articles.

References

[1] Page, L., Brin, S., Motwani, R., Winograd, T.: In: The PageRank Citation Ranking: Bringing Order to the Web. In: *Technical report*, Stanford Digital Library Technologies Project, 1998

[2] Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: *7th International World Wide Web Conference (WWW 1998)*, 1998

[3] Broder, A. et al.: Graph Structure in the Web: Experiments and Models, In: *9th International World Wide Web Conference (WWW 2000)*, Amsterdam, 2000

[4] Kropf, P.; Plaice, J.; Unger, H.: Towards a Web Operating System. In: *S. Lobodzinski and I. Tomek: Proceedings of the World Conference of the WWW, Internet and Intranet (WebNet'97)*, pp. 994–995, Toronto (CA), 1997

[5] Sodsee, S.: *Placing Files on the Optimal Nodes of Peer-to-peer Systems*. Dissertation, FernUniversität in Hagen, Fortschritt-Berichte VDI, Series 10: Informatik/Kommunikation, Vol. 816, VDI, Düsseldorf, 2012

[6] Pourebrahimi, B., Bertels, K., Vassiliadis, S.: A survey of peer-to-peer networks. In: *Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Proessing*, 2005

[7] Ritter, J.: Why Gnutella Can't Scale. No, Really. In: *Technical Report*, 2001, online version at `http://www.cs.rice.edu/~alc/old/comp520/papers/ritter01gnutella-cant-scale.pdf`, Last retrieved on: 06/22/2015

[8] Milgram, S.: The Small World Problem. In: *Psychology Today*, Vol. 2, pp. 60–67, 1967

[9] Castro, M.; Costa, M.; Rowstron, A.: Peer-to-peer overlays:structured, unstructured, or both. In: *Technical Report MSR-TR-2004-73*, Microsoft Research, System and Networking Group, Cambridge (UK), 2004

[10] Unger, H.; Wulf, M.: Cluster-building in P2P-Community Networks. In: *Journal on Parallel and Distributed Computing Systems and Networks*, Vol. 5(4), pp. 172–177, 2002

[11] Sakarian, G.; Unger, H.: Influence of Decentralized Algorithms on the Topology Evolution of Distributed P2P Networks. In: *Proceedings of Design, Analysis and Simulation of Distributed System (DASD) 2003*, pp. 12–18, Orlando, Florida, 2003

[12] Coltzau, H.: *Dezentrale Netzwelten als Interaktions- und Handelsplattformen.* Dissertation, FernUniversität in Hagen, 2012, online version at `http://deposit.fernuni-hagen.de/2903/`, Last retrieved on: 06/22/2015

[13] Google Autocomplete, Web Search Help, `http://support.google.com/websearch/answer/106230?hl=en`, 2015, Last retrieved on: 06/22/2015

[14] Kubek, M., Unger, H., Loauschasai, T.: A Quality- and Security-improved Web Search using Local Agents, In: *Intl. Journal of Research in Engineering and Technology (IJRET)*, Vol. 1 No. 6, 2012

[15] Kubek, M., Witschel, H. F.: Searching the Web by Using the Knowledge in Local Text Documents. In: *Proceedings of Mallorca Workshop 2010 Autonomous Systems*, Shaker Verlag Aachen, 2010

[16] Kubek, M.: DocAnalyser - Searching with Web Documents, In: *Autonomous Systems 2014, Fortschritt-Berichte VDI*, Vol. 10 Nr. 835, pp. 221–234, VDI-Verlag Düsseldorf, 2014

[17] Kubek, M., Unger, H., Hussein, P., Tiranalinvit, W., Chatpaiboonwat, P.: Contextual Rearrangement of Web Content, In: *Proceedings of the 3rd International Conference on IT and Intelligent Systems (ICITIS'2013)*, Bangkok, 2013

[18] Sukjit, P., Kubek, M., Böhme, T., Unger, H.: PDSearch: Using Pictures as Queries, In: *Recent Advances in Information and Communication Technology*, Advances in Intelligent Systems and Computing, Vol. 265, pp. 255–262, Springer International Publishing, 2014

[19] Kubek, M., Unger, H., Dusik, J.: Correlating Words - Approaches and Applications, In: *Proceedings of the 16th International Conference on Computer Analysis of Images and Patterns (CAIP 2015)*, Springer International Publishing, 2015

[20] YaCy, `http://www.yacy.de/de/index.html`, 2015, Last retrieved on 06/22/2015

[21] Faroo, `http://www.faroo.com/`, 2015, Last retrieved on 06/22/2015

[22] Meng, W., Yu, C., Liu, K.: Building efficient and effective metasearch engines. In: *ACM Computing Surveys (CSUR)*, Volume 34, Issue 1, pp. 48–89, ACM, 2002

[23] Dogpile, `www.dogpile.com/`, 2015, Last retrieved on 06/22/2015

[24] Baeza-Yates, R.: The impact of caching on search engines. In: *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07)*, pp. 183–190, ACM, 2007

[25] Peters, I.: *Folksonomies: Indexing and Retrieval in Web 2.0*, de Gruyter, 2009

[26] Wang, X., Halang, W.A.: *Discovery and Selection of Semantic Web Services*. Studies in Computational Intelligence, Volume 453, Springer Berlin Heidelberg, 2013

[27] Sriharee, G.: An ontology-based approach to auto-tagging articles. In: *Vietnam Journal of Computer Science*, Volume 2, Issue 2, pp. 85–94, Springer Berlin Heidelberg, 2014

[28] Carrot^2, `http://project.carrot2.org/`, 2015, Last retrieved on

06/22/2015
[29] Heyer, G., Quasthoff, U., Wittig, Th.: *Text Mining - Wissensrohstoff Text*, W3L Verlag Bochum, 2006
[30] Helbig, H.: *Knowledge Representation and the Semantics of Natural Language*, Springer Berlin Heidelberg, 2006
[31] Hawkins, J.: *Die Zukunft der Intelligenz.* Rowohlt Publisher, ISBN 3 499 62167 3, Hamburg, 2006