Analyzing User Behavior to Rank Desktop Items

Paul - Alexandru Chirita and Wolfgang Nejdl
L3S and University of Hannover
Deutscher Pavillon Expo Plaza 1
30539 Hannover, Germany
{chirita,nejdl}@l3s.de

December 20, 2005

Abstract

Existing desktop search applications, trying to keep up with the rapidly increasing storage capacities of our hard disks, are an important step towards more efficient personal information management, yet they offer an incomplete solution. While their indexing functionalities in terms of different file types they are able to cope with are impressive, their ranking capabilities are basic, and rely only on TFxIDF measures, comparable to the first generation of web search engines. In this paper we propose to connect semantically related desktop items by exploiting usage information about single accesses or sequences of accesses to local resources. We investigate and evaluate in detail the possibilities to translate this information into a desktop linkage structure, and we propose several algorithms that exploit these newly created links in order to efficiently rank desktop items. Finally, we empirically show that the access based links lead to ranking results comparable with TFxIDF ranking, and clearly surpass TFxIDF when used in combination with it, making them a very valuable source of input to desktop search ranking algorithms.

1 Introduction

The capacity of our hard-disk drives has increased tremendously over the past decade, and so has the number of files we usually store on our computer. Using this space, it is quite common to have over 100,000 indexable items on the desktop. It is no wonder that sometimes we cannot find a document anymore, even when we know we saved it somewhere. Ironically, in quite a few of these cases nowadays, the document we are looking for can be found faster on the World Wide Web than on our personal computer.

In view of these trends, resource organization in personal repositories has received more and more attention during the past few years. Several projects have started to explore search and resource organization on the desktop, including Stuff I've Seen [11], Haystack [27], Gnowsis [30], and our Beagle^{++} project

[10]. Moreover, at a recent panel at WWW 2005, one of the panelists identified desktop search as one of the future search engine development directions, together with other "vertical" search areas, such as image or video search [26].

Web search has become more efficient than PC search due to the powerful link based ranking solutions like the PageRank algorithm [25] introduced by Google[1]. The recent arrival of desktop search applications, which index all data on a PC, promises to increase search efficiency on the desktop. However, even with these tools, searching through our (relatively small set of) personal documents is currently inferior to searching the (rather vast set of) documents on the web. Indeed, desktop search engines are now comparable to first generation web search engines, which provided full-text indexing, but only relied on textual information retrieval algorithms to rank their results.

The main problem with ranking on the desktop comes from the lack of links between documents, the foundation of current ranking algorithms (in addition to TFxIDF measures). The semantic links offer the missing ingredients: By gathering and explicitly representing semantic information and relationships from user activities and the contexts the user works in, we construct the necessary links between documents, as well as other information useful for ranking desktop search results.

While we explored in previous work how to generate metadata and link information from specific user activities such as email or web browsing [9], in this paper we focus on sequences of information accesses to desktop resources. We investigate and evaluate in detail the possibilities to translate this information into a desktop linkage structure, and we propose several algorithms that exploit these newly created links in order to efficiently rank desktop items. Finally, we empirically show that the access based links lead to ranking results comparable with TFxIDF ranking, and clearly surpass TFxIDF when used in combination with it, making them a very valuable source of input to desktop search ranking algorithms.

This paper is ogranized as follows: We start with a discussion of the related work in Section 2. Then, in Section 3.1 we investigate the usage patterns specific to the PC desktop environment. Section 3.2 presents the desktop ranking algorithms we propose, which we then experimentally evaluate in Section 4. Finally, we conclude and discuss further work in Section 5.

2 Previous Work

Though *ranking* plays an important role on the Web, we are not aware of approaches specifically aiming at *ranking* desktop search results. More, even though there exist quite a few systems organizing personal information sources and improving information access in these environments, few of the papers describing them concentrate on search algorithms. On the other hand, the difficulty of finding desktop information has generated a rapidly increasing interest of the industry in this area, thus resulting in several releases of various desktop

[1] http://www.google.com

search applications during the past months. We will thus start this section with a brief description of some of these commercial tools, and then we will present research projects having the broader goal of organizing desktop data.

The second part of this section will discuss work related to the *algorithms* we use to infer a desktop ranking scheme. More specifically, we first present several user studies performed at the desktop level in order to understand people's behavior in such an environment. Then, as we attempt to exploit this usage analysis information to infer semantic connections between desktop items, several algorithms from the area of Semantic Web research are also relevant to our work and we thus discuss them in the end of this section.

2.1 Desktop Search Applications

Desktop search applications are not new to the industry. Only the high interest in this area is new. For example, applications such as Enfish Personal[2] have been available since 1998, usually under a commercial license. As the amount of searchable desktop data has reached very high values and will most probably also amplify in the future, the major search engines have given more focus to this area than the academia. Thus, several desktop search distributions have been released for free (e.g., Google Desktop Search[3], MSN Desktop Search[4], etc.). More, some providers have even integrated their desktop search tool into the operating system, such as Apple[5]. Finally, the open source community has also manifested its interest in this area, the most prominent approaches being Gnome Beagle[6] (now also integrated into SuSE) and KDE KAT[7], developed within the Mandriva community. Many other commercial desktop search applications exist (e.g., Copernic, Yahoo! Desktop Search, X1, Scopeware Vision, PC Data Finder, etc.), but as our main focus in this paper is to devise a desktop ranking algorithm, rather than an entire application, we will not dive into the particularities of each of these tools.

Most of the above mentioned applications target a very exhaustive list of indexed file types, including any metadata associated to them. Also, they update their index on the fly, thus inherently tracking any kind of user activity. However, all of them seem to only employ a TFxIDF measure to rank search results, without completely exploiting the usage information they have available. Thus, they inherently miss the contextual information often resulting or inferrable from explicit user actions or additional background knowledge.

2.2 Desktop Organization Systems

At present, there is no ranking algorithm specifically designed for the search task performed within the PC desktop environment. However, several *systems*

[2] http://www.enfish.com/
[3] http://desktop.google.com/
[4] http://toolbar.msn.com/
[5] http://www.apple.com/macosx/features/spotlight/
[6] http://www.gnome.org/projects/beagle/
[7] http://kat.mandriva.com/

have been constructed in order to facilitate re-finding of various stored resources. Stuff I've Seen [11] for example provides a unified index of the data that a person has seen on her computer, regardless of its type. Contextual cues such as time, author, thumbnails and previews can be used to search for and present information, but no desktop specific ranking scheme is investigated. Similarly, MyLifeBits [13] targets storing locally all digital media of each person, including documents, images, sounds and videos. They organize these data into collections and, like us, connect related resources with links. However, they do not investigate building desktop ranking algorithms that exploit these links, but rather use them to provide contextual information.

Haystack [1] emphasizes the relationship between a particular individual and her corpus. It is quite similar to our approach in the sense that it automatically creates connections between documents with similar content and it exploits usage analysis to extend the desktop search results set. Still, just like the previous articles, it does not investigate the possibilities to *rank* these results, once they have been obtained. Its follow-ups [18, 21] further explore the efficient organization of desktop resources. They use an RDF database to store metadata about the various personal items, as well as about any connections between different desktop data. Finally, Magnet [31] was designed as an additional component of Haystack with the goal to support naïve user navigation through structured information via a domain-independent search framework and user interface.

Lifestreams [12] is an older desktop organization system based a time-ordered stream of documents meant to replace conventional files and directories. All its aspects, including query results, consist of substreams of the main desktop usage stream, thus being rather different from the systems nowadays.

Hull and Hart [17] modified conventional PC peripherics (e.g., printers) to automatically store every processed document, thus providing search through any previously accessed document. They also use only traditional ranking techniques, such as ordering by date or TFxIDF. Though it does not describe the architecture of a system, the work of Ringel et al. [29] is also quite relevant for desktop search applications: They suggested using timeline visualizations augmented with public and personal landmark events in order to display query results over an index of personal content.

Finally, in prior work [9, 10] we described our Beagle^{++}[8] personal information system, a semantically enriched extension of the Beagle open sources desktop search engine. There, we proposed various heuristics to generate ample activity based metadata associated to each desktop item. In addition, we generated links between resources in a similar manner to Haystack (e.g., between a file and an email, if the former resource was stored from the attachment of the latter), and we applied a schema based PageRank [4] to compute reputation scores. In this paper we take a different approach: First, we no longer focus on the system, but on an algorithm designed for a specific task (i.e., desktop ranking); second, we explore new sources of linkage information between desktop items, such as global file access patterns.

[8] http://beagle.l3s.de/

2.3 Desktop Usage Analysis

Desktop usage behavior has been thoroughly analyzed, not only to extend the results set for desktop search queries, but also to improve user interfaces, lab communication or work performance. Malone [24] used interviews to analyze the way professional and clerical office workers organize information in their desks and offices. He identified two broad types of persons, *filers*, who organize their data into directories and categories, and *pilers*, who simply store all files in as few directories as possible. His work is orthogonal to ours, as we also analyze desktop user activity, but we focus on file access distribution, rather than storage behavior. Later, Barreau and Nardi [5] suggested that the way information is used on the PC desktop should also be the primary determinant of the way it will be organized, stored and retrieved in the personal workspace. Jones et al. [20] investigate the methods people use in their workplace to organize web information for re-use (e.g., send email to self or others, print out the web page, etc.) and Ringel [28] analyzed the way virtual desktops and multi-monitor systems are used. She pointed out that while many operating systems group all documents of the same type together on the taskbar, it may be beneficial if documents related to a particular task were grouped together instead.

Another quite different stream of research has been dedicated to endowing computing systems with the ability to sense and reason about human attention, thus improving the human-computer interaction process. Horvitz et al. [16] use perceptual sensors (e.g., microphones, camera, etc.) to train Bayesian models to reveal current or future attention under uncertainty from an ongoing stream of clues. Many other similar works exist, for example that of Horvitz and Apacible [15], who present methods for building attentional focus and workload models that can be used to infer a user's state of interruptability from multiple event sources and provide a well-characterized expected cost of interruption. As we currently only use straightforward heuristics to infer the connectivity between desktop resources, all these applications can be used to further clean the noisy links we might generate by exploiting each activity model.

2.4 Relevant Search Algorithms

Aleman-Meza et al. [2] analyzed the importance of semantically capturing users' interests in order to develop a ranking technique for the large number of possible semantic associations between the entities of interest for a specific query. They define an ontology for describing the user interest and use this information to compute weights for the links among the semantic entities. The approach is orthogonal to ours, as we build links only by exploiting fast usage analysis information, instead of using various complex algorithms to connect at run-time the desktop entities relevant for every specific user query. Another similar technique for ranking semantic query results is to analyze the inferencing processes that led to each result [32]. In this approach, the relevance of the returned results is computed slightly faster than in the previous one, by exploiting the specificity of the relations residing within the knowledge base. The calculation of the

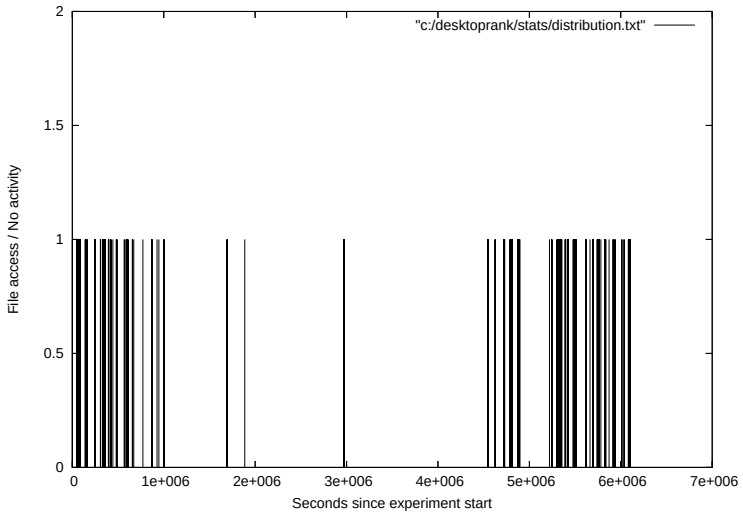

Figure 1: Two months distribution of manual accesses to desktop items.

relevance is however a problem-sensitive decision, and therefore task oriented strategies should be developed for this computation.

TAP [14] is a platform for publishing and consuming data from the Semantic Web. Its key idea is that for specific searches, a lot of information is only available in catalogs and backend databases, and not on the Web pages crawled by search engines. Thus, the results obtained via traditional information retrieval technologies can be augmented with semantic information (e.g., the photo of a person, etc.) obtained from the pre-compiled TAP knowledge base.

3 Ranking Desktop Resources

As the number of files on our desktops can easily exceed 100,000, we can no longer manage them manually just by defining "good" file and directory names and structures. More, the currently employed textual information retrieval measures such as TFxIDF are no longer sufficient to order the usually several hundreds of results returned for our desktop search queries. We therefore need to investigate more advanced desktop organization paradigms and ranking algorithms. In this section, we address the latter issue. As there is almost no previous work on analyzing basic file access activity on the PC desktop, we start by investigating the possible distributions characterizing it. Based on this investigation, we propose several algorithms that exploit file access information in order to efficiently rank desktop search results.

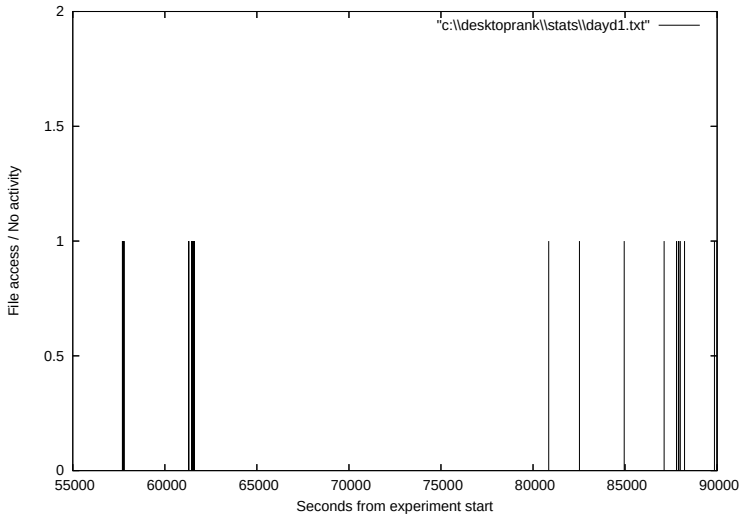

Figure 2: One day distribution of manual accesses to desktop items.

3.1 Usage Patterns on the PC Desktop

Our ultimate goal is to infer links from desktop resources that have been accessed in a sequence. Yet this is not a straightforward task. Several persons might have quite different usage behavior strategies, thus making it very difficult to distinguish usage sessions from each other. More, this problem could even occur with the same person, at two different moments in time. We thus conducted a usage behavior study with seven computer science graduate and undergraduate students in order to analyze their file access patterns. All of them manifested rather similar usage patterns on the long term. We depict in Figure 1 one user activity captured over a period of two months. We notice that on such a long term the daily accesses are rather easy to distinguish: Each bar represents one file access. When several accesses occur at small intervals, their associated bars are merged into a thicker one. Also, longer breaks have been generated during week-ends, and the three very large pauses represent vacation periods. But what happens with the activity performed during a single day? A general example is presented in Figure 2. There are two access intensive working sessions in the morning, and *only one* session in the afternoon. In general, we distinguished two broad types of desktop activity: *working* (e.g., reading an article, writing a program, etc.), which usually results in a relatively small file access frequency, and *browsing* (e.g., reading emails, surfing the web, etc.), when much more resources are opened in a short amount of time, in many cases only for reading. Therefore, we investigated three approaches to identify usage sessions:

1. Session is over after one hour of break. This is rather naïve, considering our findings above, but it is the fastest to compute.
2. Session is over after a break four times longer than the average break time of the user.

3. Session is over after a break eight times longer than the average break time of the user.

The second heuristic yielded the best results, while also being quite similar to the third one (more details can be found in the experiments Section 4. We thus used it to identify usage sessions for the other ranking algorithms we proposed.

Having identified the file access patterns, we then investigated the distributions of file access frequency and total file access time, as they represent a good indicator of how the final ranking distributions will look like. The former distribution has also been investigated by Dumais et al. [11], obtaining similar results. However, they only looked at the resources that have been accessed at least once, whereas we considered all desktop items in our analysis. This helped us obtain an additional interesting result, namely that only about 2% of the desktop indexable items are actually manually accessed by the user. This is most probably because of various program documentations (especially when in HTML format), locally stored mailing list archives, etc. We think this finding further supports the idea of exploiting usage information in ranking desktop search results, as current textual measures (i.e., TFxIDF) many times output high scores for such automatically deployed documents that have never been touched by the user. We depict a sample visit frequency distribution in Figure 3. For all our testers, this distribution followed a power law (i.e., $f(x) = c \cdot 1/x^{\gamma}$) with very low values for the γ exponent, ranging from -0.26 to -0.38. Similarly, we depict the distribution of total time spent accessing each file (i.e., reading, writing, etc.) in Figure 4. This time, the distribution can be tailored by a power law with an exponential cut-off, as in the following formula:

$$f(x) = c \cdot \frac{1}{x^{\gamma}} \cdot e^{\frac{-x}{z_c}} \tag{1}$$

The additional inverse exponential term is only used to ensure a faster decreasing value of f, z_c being a parameter. Again, we obtained very low values for γ, residing around 0.18.

It has been shown previously that the distribution of words over various collections of documents, including the World Wide Web, also follows a power-law [23]. Though this is less relevant for the ranking algorithms per se, we think that moving this word distribution analysis down to the PC desktop level provides a good insight into how the desktop search algorithms should be designed. Not surprisingly, this also yielded a scale free distribution (see Figure 5).

3.2 Desktop Ranking Algorithm

Exploiting Usage Analysis to Generate Ranks. Existing personal information systems create links between desktop resources only when a very specific desktop usage activity is encountered (e.g., the attachment of an email is saved as a file, or a web page is stored locally, etc.). We argue that in fact in almost all cases when two items are touched in a sequence several times, there will also be a relation between them, irrespective of the underlying user activity (e.g., surfing the web, etc.). Thus, we propose to add a link between such two items

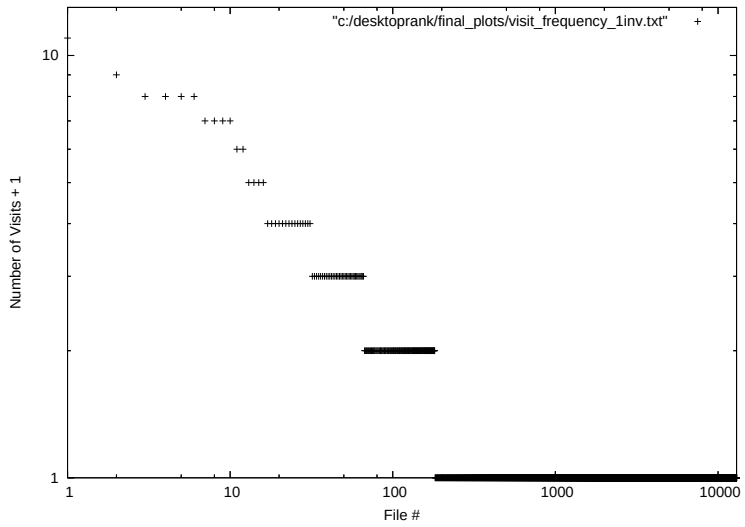

Figure 3: Frequency distribution of number of manual accesses to desktop items.

a and b whenever item b is touched after a for the T^{th} time, with T being a threshold set by the user. Higher values for T mean an increased accuracy of the ranking algorithm, at the cost of having a score associated to less resources[9].

After a short period of time a reputation metric can be computed over the graph resulted from this usage analysis process. There exist several applicable metrics. The most common one is PageRank [25]. On the one hand, it has the advantage of propagating the inferred semantic similarities (connections), i.e., if there is a link between resources a and b, as well as an additional link between resources b and c, then with a relatively high probability we should also have a connection between a and c. On the other hand, PageRank also implies a minimal additional computational overhead, which is not necessary for a simpler, yet more naïve metric, *in-link count*. According to this latter approach, the files accessed more often get a higher ranking. In-link counting has already been proved to be relatively effective even in the context of the web [7]. However, our experiments from Section 4 will show that although it does indeed yield a clear improvement over simple TFxIDF, in-link (or frequency) count is also visibly less effective than PageRank.

Another aspect that needs to be analyzed is the type of links residing on the PC desktop. In our approach we use directed links for each sequence $a \rightarrow b$, because if file b is relevant for file a, it does not necessarily mean that the reversed is true as well. Imagine for example that b is a report we are regularly appending, whereas a is the article we are writing. Clearly b is more relevant for a, than a is for b. However, since one might also argue that these "semantic" links are in

[9]Theoretically, there is only a very low probability to have any two items a and b touched in a sequence even once. However, since context switching occurs quite often nowadays, we also investigated higher values than 1 for T.

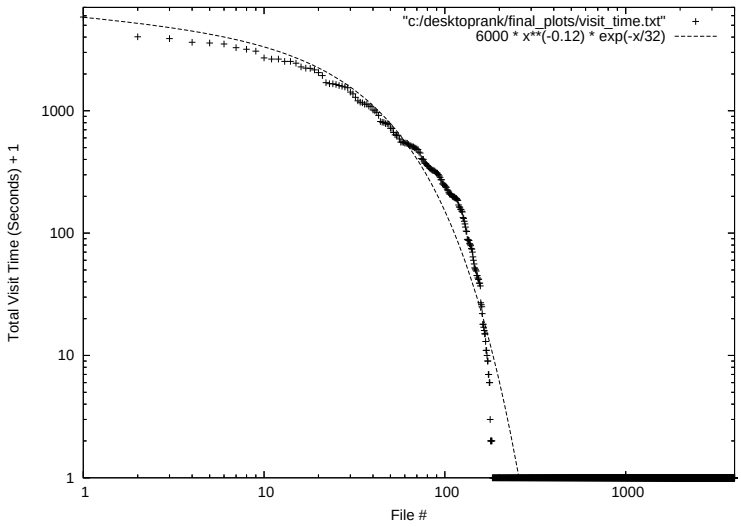

Figure 4: Distribution of time spent while manually accessing desktop items.

fact valid in both directions (possibly with different levels of importance), we intend to investigate this approach in future work[10]. We summarize the details of our proposed technique in Algorithm 3.1.

Algorithm 3.1. Ranking Desktop Items.

Pre-processing:
1: **Let** A be an empty link structure
2: **Repeat** for ever
3: **If** (File a is accessed at time t_a) AND
$(t_a - t_b < MaxSessionLen)$,
where t_b is the access time of the subsequently touched item
4: **Then** Add the link $a \rightarrow b$ to A

Ranking:
1: **Let** A' be an additional, empty link structure
2: **For** each resource i
3: **For** each resource j linked to i
4: **If** $(\#Links(i \rightarrow j) > T)$
5: **Then** Add one link $i \rightarrow j$ to A'
6: **Run** PageRank using A' as underlying link structure

---

[10]Using bi-directional links instead of uni-directional also brings the additional advantage of an easier computation of the stationary distribution of the Markov chain associated to the desktop linkage graph.

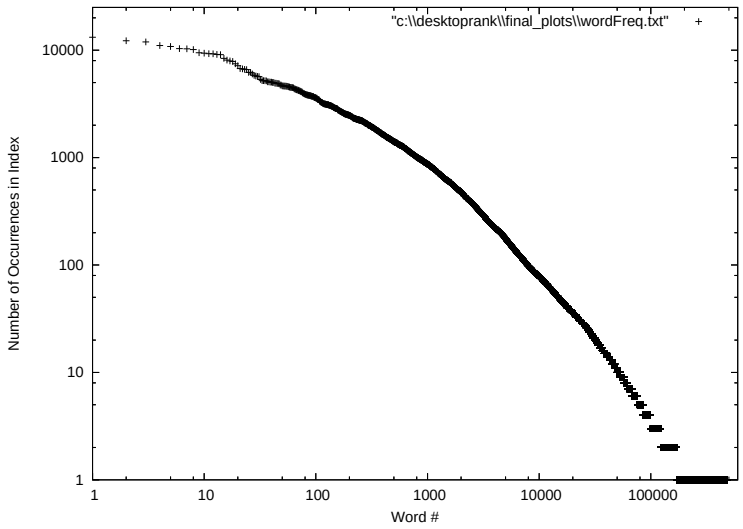

Figure 5: Frequency of words appearances within a desktop-based index.

As it was not clear how many times two resources should be accessed in a sequence in order to infer a "semantic" connection between them, we studied several values for the T threshold, namely one, two and three. Additionally, we also explored the possibilities to directly use the original matrix A with PageRank, thus implicitly giving more weight to links that occurred more frequently (recall that in A each link is repeated as many times as it occurred during regular desktop activity). Finally, in order to address a broad scope of possible ranking algorithms, we also experimented with trivial reputation measures, namely (1) frequency of accesses and (2) total access time.

Other Heuristics to Generate Desktop Links. There exists a plethora of other cues for inferring desktop links, most of them being currently unexplored by previous work. For example the *files stored within the same directory* have to some extent something in common, especially for filers, i.e., the users that organize their personal data into carefully selected hierarchies. Similarly, *files having the same file name* (ignoring the path) are in many times semantically related. In this case however, each name should not consist exclusively of stop words. More, for this second additional heuristic we had to utilize an extended stopword list, which also includes several very common file name words, such as "index", or "readme". In total, we appended 48 such words to the original list. Finally, we note that both these above mentioned approaches favor lower sets: If all files within such a set (e.g., all files residing in the same directory) are linked to each other, then the stationary probability of the Markov chain associated to this desktop linkage graph is higher for the files residing in a smaller set. This is in fact correct, since for example a directory storing 10 items has most probably been created manually, thus containing files that are to some extent related, whereas a directory storing 1,000 items has in most of the situations

been generated automatically. Also, since these sub-graphs of the main desktop graph are cliques, several computational optimizations are possible (as well as necessary); however, in order to keep our algorithms clear we will not discuss them here.

A third source of linkage information is *file type*. There is clearly a connection between the resources sharing the same type, even though it is a very small one. Unfortunately, each such category will nowadays be filled with up to several thousands of items (e.g., JPG images), thus making this heuristic difficult to integrate into the ranking scheme. A more reliable approach is to *use text similarity to generate links between very similar desktop resources*, e.g., as returned using the algorithm from [8]. Likewise, if *the same entity appears in several desktop resources* (e.g., Hannover appears both as the name of a folder with pictures and as the subject of an email), then we argue that some kind of a semantic connection exists between the two resources. Finally, we note that users should be allowed to manually create links as well, possibly having a much higher weight associated to these special links.

Practical Issues. Several special cases might arise when applying usage analysis for desktop search. We discuss here two of them. First, the textual log file capturing usage history should persist over system updates in order to preserve the rich linkage information. In our experiments, we collected only about 80 KB of log data over two months (however, this value might be a bit higher, as we only indexed several user defined paths, accounting on average for about one third of all indexable data). Second and most important, what if the user looks for a file she stored five years ago, when she had no desktop search application installed? We propose several solutions to this:

1. The naïve approach is to simply enable ranking based exclusively on TFxIDF. However, much better results can be obtained by incorporating contextual information within the ranking scheme.
2. We therefore propose a more complex query term weighting scheme, such as BM25 [19]. Teevan et al. [33] have recently proposed an application of this metric to personalize web search based on desktop content. In our approach, their method must be adapted to personalize *desktop* search based on a specific *activity context*, represented for example by the files with a specific path or date range.
3. If the user remembers the approximate moment in time when she accessed the sought item, then this date represents an useful additional context based vertical ranking measure. For example, if the user remembers having used the target file around year 1998, the additional importance measure is represented by the normalized positive time difference between 01.07.1998 (mid-1998) and the date of each output result.
4. If no contextual information is available, we propose to infer it through a relevance feedback process, in which the user first searches the desktop using TFxIDF exclusively, and then selects one or several (relatively) relevant results, which are then used to extract a context (e.g., date range) or to propose expansions to the user query.

4 Experimental Results

We evaluated the utility of our algorithms within three different environments: our laboratory (with researchers in different computer science areas and education), DFKI, a partner laboratory with slightly different computer science interests, and the architecture department of our university. The last location was especially chosen to give us an insight from persons with very different activities and requirements. In total, 11 persons installed our tool and worked normally on their desktops for two months. Then, during the following three weeks, they performed several desktop searches related to their regular activities[11], and graded each top 10 result with a score from one to five, one defining a very poor result with respect to their desktop data and expectations, and five a very good one. This is in fact a weighted *precision* at the top 10 results [3]. For every query, we shuffled the top ten URIs output by each of our algorithms, such that the users were neither aware of their actual place in the rank list, nor of the algorithm(s) that produced them. In the following, we will first present an analysis of this experiment focused on the ranking algorithms, and then another one, focused on the quality of the search output they produced.

Ranking analysis. In the first phase of our experiments, we wanted to learn how our algorithms perform, to tune the parameters discussed in the previous section, and to investigate whether the non-usage analysis heuristics we proposed do indeed make a difference in the overall rankings. We thus defined and analyzed the following 17 algorithms:

- T1: Algorithm 3.1 with $T = 1$.
- T1Dir: Algorithm 3.1 with $T = 1$, as well as additional links created as complete subgraphs with the files residing in every desktop directory (i.e., all the files in a directory point to each other).
- T1DirFnames: "T1Dir" with further additional links created as complete subgraphs with the resources having the same file name (i.e., all items with the same file name point to each other, provided that the file name does not consist exclusively of stopwords).
- T1Fnames: "T1" enriched with the links between resources with identical file names as in the previous algorithm[12]. This was necessary to inspect the specific contribution of directories and file names respectively to the overall ranking scheme.
- T1x3Dir: Same as "T1Dir", but with the links inferred from usage analysis being three times more important than those inferred from the directory structure.
- T1x3DirFnames: Same as above, but also including the links provided by identical file names.

[11] The only requirement we made here was to perform at least five queries, but almost every subject provided more. In all cases, we collected the average rating per algorithm for each person.

[12] For emails, this correspdonded to having the same subject, eventually with "Re:" or "Fwd:" inserted in the beginning.

- T1x3Fnames: Same as "T1x3Dir", but using the file name heuristic instead of the directory one.
- T2: Algorithm 3.1 with $T = 2$
- T3: Algorithm 3.1 with $T = 3$
- Freq: Ranking by access frequency
- 1HourGap: Ranking by total amount of time spent on accessing each resource, with sessions delimited by one hour of inactivity. This technique together with two subsequent ones were used to identify the best approach to demarcate usage sessions.
- 4xAvgGap: Ranking by total access time, with sessions delimited by a period of inactivity longer than four times the average time spent using a resource. This metric had the highest precision in determining user activity sessions.
- 8xAvgGap: Same as above, but with sessions bounded by a period of inactivity longer than eight times the average item access duration.
- Weighted: Algorithm 3.1 directly using the matrix A, instead of A', i.e., with each link weighted by the number of times it occurred.
- WeightedDir: Algorithm "Weighted" enriched with links between the files stored within the same directory.
- WeightedDirFnames: The previous algorithm with a link structure extended with connections between files with identical names.
- WeightedFnames: Same as above, but without the links generated by exploiting the desktop directory structure.

Since in-link count is almost identical to file access count (frequency), we only experimented with the latter measure. The only difference between these two measures is that in-link count will result in lower page scores when a threshold higher than one is used to filter-out the links (see also Algorithm 3.1).

We analyzed two aspects at this stage: First, it was important to inspect the final distribution of rankings, as this indicates how desktop search output looks like when using these algorithms. In *all* cases the resource rankings exhibits a distribution very well shaped by a power law (see Figure 6 for an example plot of the output rankings, using the "T1" algorithm). When the directory and the file name heuristics were added, the overall distribution still followed a power law, with a strong exponential cut-off towards the end, for the files that benefited less from the link enhancement techniques. We plot this final distribution in Figure 7[13].

The second aspect to analyze was whether there is a difference between these heuristics. For this purpose we used the following version of Kendall's τ measure of similarity between two ranking vectors, resulting in a similarity score falling in [-1,1] (for more details, see Kendall [22] or Boldi et al. [6]):

Definition 1 *Let $\mathbf{r}, \mathbf{s} \in R$ be two rankings. Given a pair of distinct indices $1 \leq i, j \leq n$ (where $n = |\mathbf{r}| = |\mathbf{s}|$), we say that the pair is:*

[13]Note that in this case the scores are more uniformly distributed over all resources.

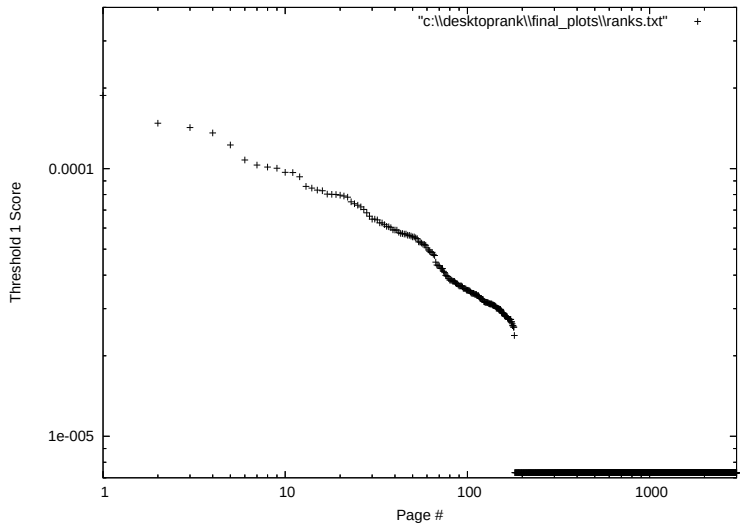

Figure 6: Distribution of scores for the "T1" algorithm.

- Concordant, *iff* $r_i - r_j$ *and* $s_i - s_j$ *are both nonzero and have the same sign;*
- Discordant, *iff* $r_i - r_j$ *and* $s_i - s_j$ *are both nonzero and have opposite signs;*
- *An* r-tie, *iff* $r_i - r_j = 0$*;*
- *An* s-tie, *iff* $s_i - s_j = 0$*;*
- *A* joint tie, *iff* $r_i - r_j = s_i - s_j = 0$*;*.

Let C, D, T_r, T_s *and* J *be the number of concordant pairs, discordant pairs, r-ties, s-ties and joint ties, respectively; let also* $N = \frac{n \cdot (n-1)}{2}$*. Of course* $C + D + T_r + T_s - J = N$*. Kendalls* τ *of the two rankings can now be defined as:*

$$\tau = \frac{C - D}{\sqrt{(N - T_r) \cdot (N - T_s)}} \tag{2}$$

Three of our testers (one from each location) were specifically asked to extensively use our tool. When they reached 40 queries each, we applied the Kendall measure on their complete output, as returned by each algorithm. The results are illustrated in Table 1. After analyzing them, we drew the following conclusions:

- The heuristics to add links between the resources residing within the same directory, or between the resources with identical file names did result in a rather different query output.

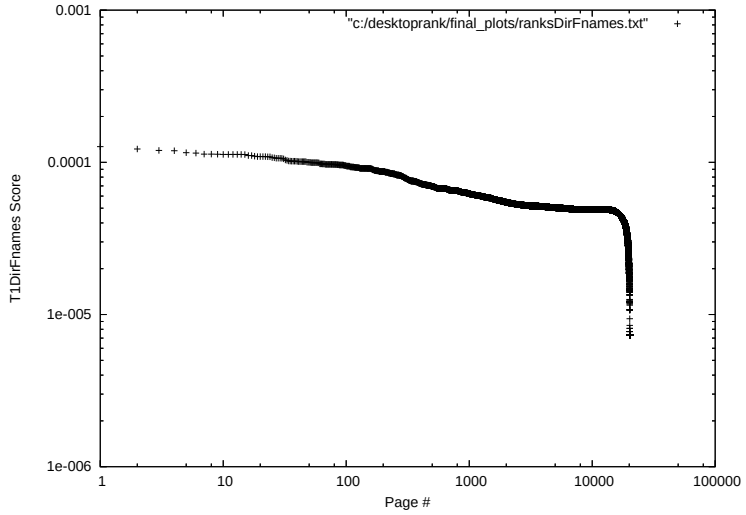

Figure 7: Distribution of scores for the "T1DirFnames" algorithm.

- The approaches "T1x3Dir", "T1x3DirFnames" and "T1x3Fnames", in which the usage analysis inferred links received three times more weight than the other heuristics (i.e., directory structure and identical file names), did not yield a significant difference in the results. We thus removed them from the search quality analysis (presented in the second half of this section).
- The output of "T2" and "T3" was very similar, indicating that a threshold higher than 2 is not necessary for Algorithm 3.1. Therefore, we did not consider "T3" in the quality analysis experiments.
- "1HourSession", "4xAvgGap" and "8xAvgGap" performed relatively similar to "T1" and even more similar to each other. Only one of them was kept for the second phase of the evaluation, namely "4xAvgGap", which we selected after a manual inspection of the output for each of the three approaches.
- "Weighted" output was very close to "T1", but we still kept it for the quality analysis, as a representative of this approach.
- Finally, when "Weighted" was combined with directory or file name information, we obtained almost identical outcomes as when we used "T1" with these heuristics. We thus considered these approaches not to yield significant differences and removed them from the second stage experiments.

Search quality analysis. Once we investigated the influence of each algorithm upon the rankings, we were ready to look into the quality of its results as well. From the previous experiment, we kept a total of eight algorithms for precision evaluation: "T1", "T1Dir", "T1DirFnames", "T1Fnames", "T2", "Freq",

"4xAvgGap" and "Weighted". Even though they do not incorporate any textual information (e.g., term frequency), we still started with ranking desktop search results only according to these measures, in order to see the impact of usage analysis on desktop ranking. The average results are summarized in Table 2 (recall that we evaluated the quality of our algorithms with 11 testers coming from three different environments). As we can see, all measures performed worse than TFxIDF (both general approaches included an implementation of Porter's stemmer to select the query hits), but only at a small difference. This indicates that users do issue a good amount of their desktop queries on aspects related to their relatively recent, or even current work. Also, as the "T2" algorithm does not improve over "T1", it is therefore sufficient to use Algorithm 3.1 with a threshold $T = 1$ in order to effectively catch the important desktop documents. This is explainable, since a threshold $T = 2$ would only downgrade files that were accessed only once, which have a relatively low score anyway compared to the other more frequently touched resources. Furthermore, in this setting the directory and file name heuristics lead to worse results, probably because they promoted items that covered the user queries only minimally (since term frequency was not considered here). Nevertheless, they perform much better when combined with TFxIDF.

Finally we investigated how our algorithms perform within a realistic desktop search scenario, i.e., combined with term frequency information. We used the following formula:

$$Score(file) = NormalizedScore(file) * \\ NormalizedVectorSpaceScore(file, query)$$

The vector space score is computed using the Vector Space Model and both scores are normalized to fall within [0,1] for a given query[14]. The resulted average grading are presented in Table 3. We notice that in this approach, *all* measures outperform TFxIDF in terms of weighted precision at the top 10 results, and most of them do that at a statistically significant difference (see column 3 of Table 3 for the statistical significance analysis with respect to each metric).

First, the usage analysis based PageRank ("T1") is clearly improving over regular TFxIDF ranking. As for the additional heuristics evaluated, connecting items with similar file name or residing in the same directory, they yielded a significant improvement only when both of them have been used. This is because when used by themselves, these heuristics tend to move the results bias away from the usage analysis information, which is the most important by far. When used together, they add links in a more uniform manner, thus including the information delivered by each additional heuristic, while also keeping the

[14]In order to avoid obtaining many null file scores when using access frequency or total access time (recall that many items have never been touched by the user), in these scenarios we also added a $1/N$ score to all items before normalizing, with N being the total amount of desktop items.

main bias on usage analysis. Finally, the simpler usage analysis metrics we investigated (e.g., ranking by frequency or by total access time) did indeed improve over TFxIDF as well, but with a lower impact than the Algorithm 3.1 enriched with directory and file name information. We conclude that with TFxIDF in place, usage analysis significantly improves desktop search output rankings and it can be further enhanced by linking resources from the same directory and with identical file names.

The final results are also illustrated in Figure 8, in order to make the improvement provided by our algorithms also visible at a graphical level. The horizontal line residing at level 3.09 represents the performance of TFxIDF; the red bars depict the average grading of the algorithms combining TFxIDF with our approaches, and the blue ones depict the average grading obtained when using only our usage analysis algorithms to order desktop search output.

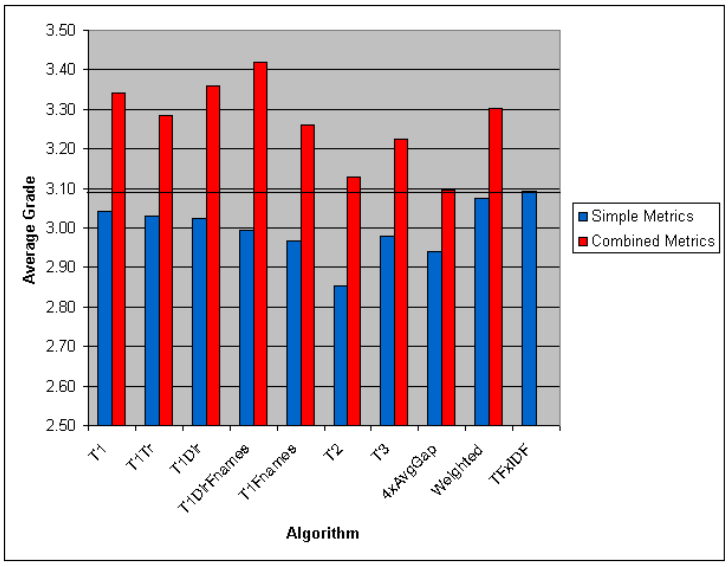

Figure 8: Average grading for the usage analysis algorithms.

5 Conclusions and Future Work

Currently there are quite several personal information systems managing PC desktop resources. However, all of them have focused on seeking solutions to find previously stored items in a faster way. In this paper we argued that in many cases these existing approaches already yield several hundreds of query results, which cannot be successfully ordered by using TFxIDF exclusively. We proposed introducing *ranking* for desktop items. We investigated in detail several approaches to achieve this goal, ranging from usage analysis to contextual

information, and through extensive experiments we showed that they indeed significanly increase desktop search quality.

Even though in this work we focused on specifically improving desktop ranking algorithms, we will also continue our work on our Beagle^{++} personal information system. Some immediate improvements, which surprisingly are *not* a part of the commercial tools we are aware of, include (1) restricting desktop search to only a specific sub-tree of the desktop hierarchy, and (2) clustering near-duplicate resources[15]. At the algorithms level, we intend to investigate further desktop connectivity heuristics, especially within the area of content based link generation.

6 Acknowledgements

First and foremost we thank Leo Sauermann from DFKI for his valuable suggestions and for implementing the module that logs user desktop activity. We also thank our colleagues Jörg Diederich and Uwe Thaden for pre-reviewing our paper, as well as to all those who kindly agreed to participate in our experiments. Finally, we thank all colleagues involved in the Beagle^{++} project for the interesting discussions we had within this context.

References

[1] E. Adar, D. Kargar, and L. A. Stein. Haystack: per-user information environments. In *Proc. of the 8th Intl. Conference on Information and Knowledge Management (CIKM)*, pages 413–422, 1999.

[2] B. Aleman-Meza, C. Halaschek, I. B. Arpinar, and A. Sheth. Context-aware semantic association ranking. In *Semantic Web and Databases Workshop*, 2003.

[3] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.

[4] A. Balmin, V. Hristidis, and Y. Papakonstantinou. Objectrank: Authority-based keyword search in databases. In *Proc. of the Conf. on Very Large Databases (VLDB)*, Toronto, Sept. 2004.

[5] D. Barreau and B. Nardi. Finding and reminding: File organization from the desktop. *ACM SIGCHI Bulletin*, 27(3):39–43, July 1995.

[6] P. Boldi, M. Santini, and S. Vigna. Do your worst to make the best:: Paradoxical effects in pagerank incremental computations. In *Workshop on Algorithms and Models for the Web Graph*, 2004.

[15] In a survey we performed with computer science and architecture students, we found that over 65% of them keep on their PCs several versions of the reports/homeworks they write, while most of the others use versioning systems such as CVS and few simply prefer having only one copy of their work, especially after this was finished.

[7] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *Proc. of the 10th Intl. Conf. on World Wide Web*, pages 415–429, 2001.

[8] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In *Proc. of the 6th Intl. World Wide Web Conference*, 1997.

[9] P. A. Chirita, R. Gavriloaie, S. Ghita, W. Nejdl, and R. Paiu. Activity based metadata for semantic desktop search. In *Proc. of the 2nd European Semantic Web Conference*, Heraklion, Greece, May 2005.

[10] P. A. Chirita, S. Ghita, W. Nejdl, and R. Paiu. Semantically enhanced searching and ranking on the desktop. In *Proc. of the 1st Workshop on The Semantic Desktop held at the 4th Intl. Semantic Web Conference*, 2005.

[11] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. Stuff i've seen: a system for personal information retrieval and re-use. In *Proc. of the 26th Intl. ACM SIGIR Conf. on Research and Development in Informaion Retrieval*, pages 72–79, 2003.

[12] S. Fertig, E. Freeman, and D. Gelernter. Lifestreams: An alternative to the desktop metaphor. In *Proc. of the ACM Conference on Human Factors in Computing Systems*, pages 410 – 411, 1996.

[13] J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: fulfilling the memex vision. In *Proc. of the ACM Conference on Multimedia*, 2002.

[14] R. Guha, R. McCool, and E. Miller. Semantic search. In *Proceedings of the 12th International World Wide Web Conference*, pages 700–709, 2003.

[15] E. Horvitz and J. Apacible. Learning and reasoning about interruption. In *Proc. of the 5th Intl. Conf. on Multimodal Interfaces*, pages 20–27, 2003.

[16] E. Horvitz, C. Kadie, T. Paek, and D. Hovel. Models of attention in computing and communication: From principles to applications. *Communications of the ACM*, 46(3), 2003.

[17] J. Hull and P. Hart. Toward zero-effort personal doc. management. *IEEE Computer*, 34(3):30–35, 2001.

[18] D. Huynh, D. Karger, and D. Quan. Haystack: A platform for creating, organizing and visualizing information using rdf. In *Proc. of the Sem. Web Workshop held at 11th World Wide Web Conf.*, 2002.

[19] K. S. Jones, S. Walker, and S. Robertson. Probabilistic model of information retrieval: Development and status. Technical report, Cambridge University, 1998.

[20] W. Jones, S. Dumais, and H. Bruce. Once found, what then?: A study of keeping behaviors in the personal use of web information. In *Proc. of ASIST*, 2002.

[21] D. R. Karger, K. Bakshi, D. Huynh, D. Quan, and V. Sinha. Haystack: A customizable general-purpose information management tool for end users of semistructured data. In *Proc. of the 1st Intl. Conf. on Innovative Data Systems Research*, 2003.

[22] M. Kendall. *Rank Correlation Methods*. Hafner Publishing, 1955.

[23] M. Levene, T. Fenner, G. Loizou, and R. Wheeldon. A stochastic model for the evolution of the web. *Computer Networks*, 39:277–287, 2002.

[24] T. Malone. How do people organize their desks? implications for the design of office information systems. *ACM Transactions on Office Information Systems*, 1(1):99–112, 1983.

[25] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.

[26] J. Pedersen. How search engines shape the web. In *14th Intl. World Wide Web Conference*, 2005.

[27] D. Quan and D. Karger. How to make a semantic web browser. In *Proc. of the 13th International World Wide Web Conf.*, 2004.

[28] M. Ringel. When one isn't enough: an analysis of virtual desktop usage strategies and their implications for design. In *CHI '03 extended abstracts on Human Factors in Computing Systems*, pages 762–763, 2003.

[29] M. Ringel, E. Cutrell, S. Dumais, and E. Horvitz. Milestones in time: The value of landmarks in retrieving information from personal stores. In *INTERACT*, Zurich, Sept. 2003.

[30] L. Sauermann and S. Schwarz. Gnowsis adapter framework: Treating structured data sources as virtual rdf graphs. In *Proceedings of the 4th International Semantic Web Conference*, 2005.

[31] V. Sinha and D. R. Karger. Magnet: supporting navigation in semistructured data environments. In *Proc. of the 2005 ACM SIGMOD Intl. Conference on Management of Data*, pages 97–106, 2005.

[32] N. Stojanovic, R. Studer, and L. Stojanovic. An approach for the ranking of query results in the semantic web. In *Proc. of the 2nd International Semantic Web Conference*, 2003.

[33] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of the 28th Intl. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 449–456, 2005.

Algorithm	T1	T1Dir	T1DirFnames	T1Fnames	T1x3Dir	T1x3DirFnames	T1x3Fnames	T2	T3	Freq	1HourGap	4xAvgGap	8xAvgGap	Weighted	WeightedDir	WeightedDirFnames
Threshold 1	1															
T1Dir	0.22	1														
T1Dir-Fnames	0.22	0.47	1													
T1Fnames	0.23	0.22	0.35	1												
T1x3Dir	0.28	0.86	0.46	0.23	1											
T1x3Dir-Fnames	0.24	0.48	0.75	0.40	0.48	1										
T1x3-Fnames	0.22	0.24	0.36	0.88	0.24	0.41	1									
Threshold 2	0.20	0	-0.2	0	0.02	-0.2	0	1								
Threshold 3	0.01	-0.1	-0.3	-0.1	-0.1	-0.3	-0.1	0.60	1							
Frequency	0.66	0.24	0.15	0.27	0.26	0.20	0.28	0.26	0.05	1						
1HourGap	0.48	0.15	0.14	0.20	0.12	0.11	0.20	0.17	0.02	0.41	1					
4xAvgGap	0.43	0.25	0.18	0.23	0.26	0.19	0.24	0.20	0.04	0.43	0.34	1				
8xAvgGap	0.48	0.26	0.16	0.21	0.27	0.18	0.22	0.16	0.04	0.50	0.47	0.70	1			
Weighted	0.75	0.20	0.21	0.20	0.25	0.24	0.20	0.24	0.01	0.64	0.52	0.47	0.47	1		
Weighted-Dir	0.22	0.89	0.47	0.22	0.85	0.48	0.24	0	-0.1	0.21	0.11	0.26	0.27	0.22	1	
Weighted-DirFnames	0.21	0.47	0.89	0.34	0.46	0.75	0.36	-0.2	-0.3	0.15	0.14	0.18	0.16	0.21	0.47	1
Weighted-Fnames	0.26	0.24	0.37	0.83	0.25	0.43	0.81	0	-0.1	0.31	0.28	0.28	0.26	0.25	0.24	0.36

Table 1: Kendall similarity for the desktop ranking algorithms (average over 120 queries from 3 users).

Algorithm	Average
T1	3.04
T1Dir	3.02
T1DirFnames	2.99
T1Fnames	2.97
T2	2.85
Freq	2.98
4xAvgGap	2.94
Weighted	3.07
TFxIDF	3.09

Table 2: Average grading for employing *only the usage analysis algorithms* when ranking desktop search results.

Algorithm	**Average**	**Statistical Significance versus TFxIDF**
T1 * TFxIDF	3.34	High, $p = 0.003$
T1Dir * TFxIDF	3.36	High, $p < 0.001$
T1DirFnames * TFxIDF	3.42	High, $p \ll 0.001$
T1Fnames * TFxIDF	3.26	Minimal, $p = 0.064$
T2 * TFxIDF	3.13	No, $p = 0.311$
Freq * TFxIDF	3.23	Minimal, $p = 0.141$
4xAvgGap * TFxIDF	3.09	No, $p = 0.494$
Weighted * TFxIDF	3.30	Yes, $p = 0.012$
TFxIDF	3.09	

Table 3: Average grading for the usage analysis algorithms combined with TFxIDF, together with tests on the statistical significance of the improvement they bring over Simple TFxIDF.