Analyzing User Behavior to Rank Desktop Items

Paul-Alexandru Chirita and Wolfgang Nejdl

L3S Research Center / University of Hanover
Deutscher Pavillon, Expo Plaza 1
30539 Hanover, Germany
{chirita, nejdl}@l3s.de

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 textual retrieval measures, comparable to the first generation of web search engines. In this paper we propose to connect semantically related desktop items by exploiting usage analysis information about sequences of accesses to local resources, as well as about each user's local resource organization structures. 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 significantly 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 some 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 years. Thus, several projects have started to explore search and personal information management on the desktop, including Stuff I've Seen [6], Haystack [13], or our Beagle^{++} [4].

Web search has become more efficient than PC search due to the powerful link based ranking solutions like PageRank [12]. 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.

F. Crestani, P. Ferragina, and M. Sanderson (Eds.): SPIRE 2006, LNCS 4209, pp. 86–97, 2006.

Desktop ranking is hindered by the lack of links between documents, an important source of evidence for current web ranking algorithms. In this paper we propose to alleviate this deficiency by analyzing user's activity patterns, as well as her local resource organization structures. 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 significantly surpass TFxIDF when used in combination with it, making them a very valuable source of input to desktop search ranking algorithms.

The paper is organized as follows: We start with a discussion of the relevant background in Section 2. Then, in Section 3 we present the desktop ranking algorithms we propose and in Section 4 we show our experimental results. Finally, we conclude and discuss further work in Section 5.

2 Relevant Background

Though *ranking* plays an important role on the Web, there is almost no approach 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. This section will describe several such systems and discuss their approaches to desktop search.

Several systems have been constructed in order to facilitate re-finding of various stored resources on the desktop. Stuff I've Seen [6] 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 [7] 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, 9] 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. However, just like the previous articles, it does not investigate the possibilities to *rank* these results, once they have been obtained.

Connections [14] is a very recent system also targeted at enhancing desktop search quality. Similar to us and to Haystack, they also attempt to connect related desktop items, yet they exploit these links using rather complex measures combining BFS and link analysis techniques, which results in rather large search response delays. Nevertheless, while our algorithms are clearly faster, we intend to compare the two approaches in terms of output quality in future work.

Finally, Chirita et al. [3, 4] proposed various activity specific heuristics to generate links between resources. There, our approach was limited to specific desktop contexts (e.g., publications, or web pages), whereas in this paper we explore much more general sources of linkage information such as file access patterns, which are applicable to *any* desktop resource.

3 Ranking Desktop Resources

Introduction. As the number of indexable items on our desktops (i.e., files that contain any kind of textual information, emails, etc.) 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 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 and propose several algorithms that exploit file access information in order to efficiently rank desktop search results.

Exploiting Usage Analysis to Generate Ranks. Current 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. Thus, we propose to add a link between such two 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. 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 for T, but experimental results showed them to perform worse than $T = 1$. This is in fact correct, since two files are accessed consequently more often because they are indeed related, than due to a switch of context.

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 [12]. 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 small 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. However, our experiments from Section 4 will show that although it does indeed yield a clear improvement over simple TFxIDF, file access counting is also significantly less effective than PageRank.

Another aspect that needs to be analyzed is the type of links residing on the PC desktop. We use directed links for each sequence $a \rightarrow b$, as when 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. This yields the following algorithm:

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 $t_a$, File $b$ is accessed at time $t_b$) AND ($t_a - t_b < \epsilon$),
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 $(\#Links(i \rightarrow j) > T)$ in $A$
5:              Then Add one link $i \rightarrow j$ to $A'$
6: Run PageRank using $A'$ as underlying link structure
```

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 more 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., 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 stopwords. 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; 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*. 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. 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. Second and more 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 [8]. Teevan et al. [15] 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 a 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 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) or to propose expansions to the user query.

Comparison to the Web Model. Clearly, unlike in the web, most of the desktop search queries are navigational: users just want to locate something they know their stored before. So, are some desktop files more important than others, or are they all approximately equally important? We argue that, *as in the web*, some desktop resources are much more important than others, and thus users will most of the time be seeking only for these highly important items. For example, one year after some project was closed, a log file inspected by the researcher 400 times during an experiment will definitely be less important than the project report which was probably accessed only 100 times. Therefore, contextual information, though very important, is not sufficient in effectively locating desktop items, and more complex importance measures are needed in order to exploit user's activity patterns, her local desktop organization, etc. We thus propose to link together the resources matching these heuristics (i.e., having similar access patterns, etc.), and then to utilize the resulting linkage structure to infer a global ranking over the PC Desktop.

4 Experimental Results

Experimental Setup. We evaluated the utility of our algorithms within three different environments: our laboratory (with researchers in different computer science areas and education), 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 logging tool and worked normally on their desktops for 2 months[1]. Then, during the subsequent 3 weeks, they performed several desktop searches related to their regular activities[2], and graded each top 10 result of each algorithm with a score ranging from 1 to 5, 1 defining a very poor result with respect to their desktop data and expectations, and 5 a very good one. This is in fact a Weighted P@10 [2]. 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. On average, for every issued query the subjects had to evaluate about 30 desktop documents (i.e., the reunion of the outputs of all approaches we investigated). In total, 84 queries had been issued and about 2,500 documents were evaluated.

For the link based ranking algorithms (recall that for the sake of completeness we have also evaluated some access time ranking heuristics) we set the parameter ϵ to four times the average break time of the user. We have also attempted to set it to one hour, and eight times the average break time of the user, but manual inspection showed these values to yield less accurate usage sessions. Although much more complex techniques for computing usage session times do exist (e.g., exploiting mouse clicks or movements, scrollbar activities, keyboard activities, document printing, etc. [5, 11]), we think this heuristic suffices for proving our hypothesis, i.e., usage analysis based ranking improves over simple textual retrieval approaches.

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. We first analyzed how our algorithms perform, in order to tune the parameters discussed before and to investigate whether the non-usage analysis heuristics 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:** “T1” enriched with 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).

[1] The logger was implemented using a hook that catched all manual file open / create / save system calls.

[2] The only requirement we made here was to perform at least 5 queries, but almost every subject provided more. In all cases, we collected the *average* rating per algorithm for each person.

- **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[3]. 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.
- **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 $T = 3$.
- **VisitFreq:** Ranking by access frequency.
- **1HourGap:** Ranking by total amount of time spent on accessing each resource, with sessions delimited by one hour of inactivity.
- **4xAvgGap:** Ranking by total access time, with sessions delimited by a period of inactivity longer than four times the average break time of the user.
- **8xAvgGap:** Same as above, but with sessions bounded by a period of inactivity longer than eight times the average average break time of the user.
- **Weighted:** Algorithm 3.1 directly using the matrix A, instead of A', i.e., with links weighted by the number of times they 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: The left side of Figure 1 plots the output rankings for algorithm "T1", and its right side depicts the output when both directory and file name heuristics were added (in this latter case we notice a strong exponential cut-off towards the end, for the files that benefited less from the link enhancement techniques).

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

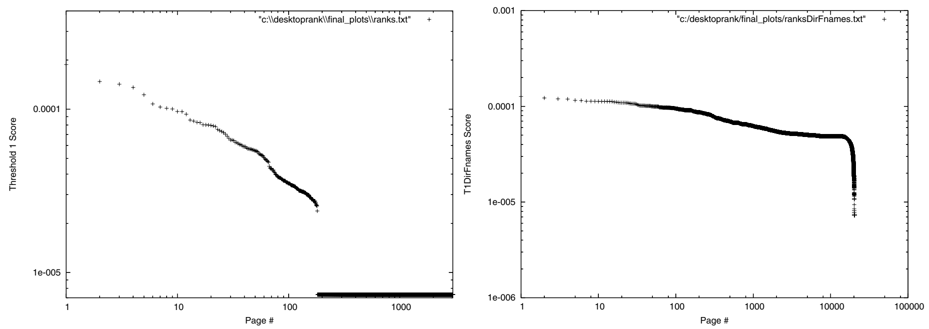

Fig. 1. Distribution of scores for the "T1" (left) and "T1DirFnames" (right) algorithms

The second aspect to analyze was whether there is a difference between these heuristics. For this purpose we used a variant of Kendall's τ measure of similarity between two ranking vectors [10], which resulted in a similarity score falling within [-1,1].

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 link the resources residing within the same directory, or the resources with identical file names did result in a rather different query output.
- The approaches "T1x3Dir", "T1x3DirFnames" and "T1x3Fnames" did not yield a significant difference in the results.
- The output of "T2" and "T3" was very similar, indicating that a threshold higher than 2 is not necessary for Algorithm 3.1.
- "4xAvgGap" and "8xAvgGap" performed very similar to each other.
- "Weighted" output was very close to "T1".
- Finally, when "Weighted" was combined with directory or file name information, we obtained almost identical outcomes as when we used "T1" with these heuristics.

As a rule of thumb, we considered similar all algorithm pairs with a Kendall τ score above 0.5, and therefore removed one of them from the search quality evaluation. Exceptions were "Weighted" and "VisitFreq" (both very similar to "T1") in order to have at least one representative of their underlying heuristics as well.

Finally, inspecting the rank distributions generated by these heuristics also 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 further supports the idea of exploiting usage information in ranking desktop search results, as current textual measures many times output high scores for documents that have never been touched by the user (e.g., HTML program documentation files).

Search quality analysis. After the previous analysis, we kept 8 algorithms for precision evaluation: "T1", "T1Dir", "T1DirFnames", "T1Fnames", "T2", "VisitFreq", "4xAvgGap" and "Weighted". Even though they do not incorporate any textual information,

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

Algorithm	T1	T1Dir	T1DirFnames	T1Fnames	T1x3Dir	T1x3DirFnames	T1x3Fnames	T2	T3	VisitFreq	1HourGap	4xAvgGap	8xAvgGap	Weighted	WeightedDir	WeightedDirFnames	WeightedFnames
Threshold 1	1																
T1Dir	0.22	1															
T1DirFnames	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											
T1x3Fnames	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								
VisitFreq	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			
WeightedDir	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	1

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 the second column of Table 2. As we can see, all measures performed worse than TFxIDF (we used Lucene[4] together with an implementation of Porter's stemmer to select the query hits, as well as to compute the TFxIDF values), 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.

[4] http://lucene.apache.org

Table 2. Average grading for the usage analysis algorithms with and without a combination with TFxIDF, together with tests on the statistical significance of the improvement the latter ones bring over regular TFxIDF

Algorithm	**Weighted P@10** (Usg. An.)	**Weighted P@10** (Combined)	**Signif. for Combined versus TFxIDF**
T1 * TFxIDF	3.04	3.34	$p = 0.003$
T1Dir * TFxIDF	3.02	3.36	$p < 0.001$
T1DirFnames * TFxIDF	**2.99**	**3.42**	$\mathbf{p \ll 0.001}$
T1Fnames * TFxIDF	2.97	3.26	$p = 0.064$
T2 * TFxIDF	2.85	3.13	$p = 0.311$
VisitFreq * TFxIDF	2.98	3.23	$p = 0.141$
4xAvgGap * TFxIDF	2.94	3.09	$p = 0.494$
Weighted * TFxIDF	3.07	3.30	$p = 0.012$
TFxIDF	3.09	3.09	

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:

$$Score(file) = NormalizedScore(file) * NormalizedVSMScore(file, query)$$

The VSM score is computed using the Vector Space Model and both scores are normalized to fall within [0,1] for a given query[5]. The resulted average gradings are presented in the third column of Table 2. 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 4 of Table 2 for the p values with respect to each metric).

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 bias the results 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 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 2, 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 (right hand side) depict the average grading of the algorithms combining TFxIDF with our approaches, and the blue ones (left hand side) depict the average grading obtained when using only our usage analysis algorithms to order desktop search output.

[5] In order to avoid obtaining many null 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.

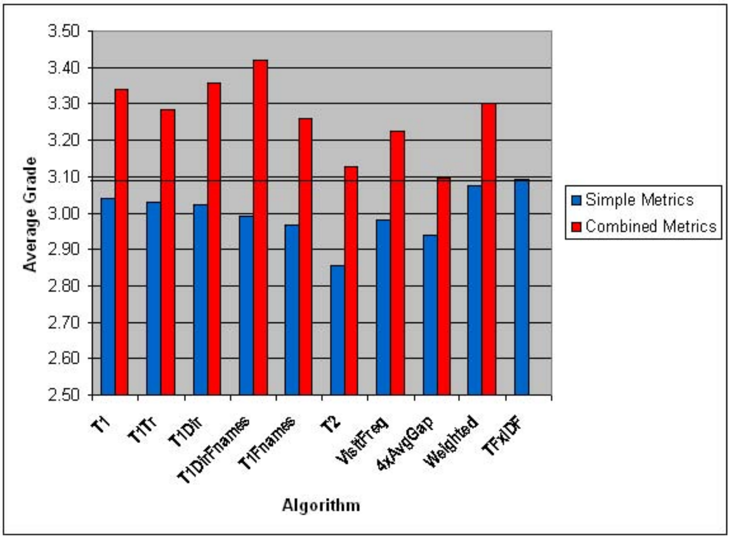

Fig. 2. Average grading for the usage analysis algorithms

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 textual retrieval measures exclusively. To solve this problem, we proposed to introduce *ranking* for desktop items and we investigated in detail several approaches to achieve this goal, ranging from usage analysis to exploiting contextual information. Our extensive experiments showed that such techniques do indeed significantly increase desktop search quality with up to 10.67% in terms of Average (Weighted) Precision.

In future work we intend to explore content based heuristics to provide us with additional links between similar desktop documents, as well as to combine our techniques with "recency" information about file accesses, which was previously proved to be quite important in locating desktop resources [6]. Also, we would like to analyze the necessity and benefits of enabling desktop search restrictions to only some specific sub-tree of the local hierarchy, as well as of clustering near-duplicate desktop resources (which is a phenomenon more common than in the web).

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. CIKM Conf. on Information and Knowledge Management*, 1999.
2. R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.
3. 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*, 2005.
4. P. A. Chirita, S. Ghita, W. Nejdl, and R. Paiu. Beagle++: Semantically enhanced searching and ranking on the desktop. In *Proc. of the 3rd European Semantic Web Conference*, 2006.
5. M. Claypool, D. Brown, P. Le, and M. Waseda. Inferring user interest. *IEEE Internet Computing*, 5(6), 2001.
6. 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.
7. 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.
8. K. S. Jones, S. Walker, and S. Robertson. Probabilistic model of information retrieval: Development and status. Technical report, Cambridge University, 1998.
9. 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 Syst.*, 2003.
10. M. Kendall. *Rank Correlation Methods*. Hafner Publishing, 1955.
11. D. Oard and J. Kim. Modeling information content using observable behavior. In *Proceedings of the 64th Annual Meeting of the American Society for Information Science and Technology*, 2001.
12. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.
13. D. Quan and D. Karger. How to make a semantic web browser. In *Proc. of the 13th Intl. WWW Conf.*, 2004.
14. C. Soules and G. Ganger. Connections: using context to enhance file search. In *SOSP*, 2005.
15. 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 Conf. on Research and Development in Information Retrieval*, 2005.