Searching a File System using Inferred Semantic Links

Deepavali Bhagwat and Neoklis Polyzotis
Computer Science Department
Univ. of California Santa Cruz
Santa Cruz, California, USA
{dbhagwat,alkis}@cs.ucsc.edu

ABSTRACT

We describe Eureka, a file system search engine that takes into account the inherent relationships among files in order to improve the rankings of search results. The key idea behind our approach is a simple, yet powerful framework that automatically infers semantic links among files and thus transforms the file system in a network of hyper-linked documents. Based on this model, we propose the FileRank metric that examines the structure of the semantic graph and essentially quantifies the "importance" of each file in the file system. By combining FileRank with conventional IR metrics, Eureka can bias the rankings of the search results toward the more important files and thus provide more effective support in the task of locating useful files. We outline the design of the Eureka search engine and discuss the inference of semantic links and the computation of the FileRank metric.

Categories and Subject Descriptors

H.5.4 [**Hypertext/Hypermedia**]: Search and Retrieval

General Terms

Links,Search

Keywords

File System, Search Engine, Ranking, Eureka

1. INTRODUCTION

Internet search engines, such as Google or Yahoo!, have been very successful in tackling a very challenging problem, namely, locating useful information on the Web. It is only natural, therefore, that researchers have looked into the development of *file system search engines*, in order to facilitate the efficient retrieval of files within large file systems. The latter is becoming an increasingly important problem, as the steady decrease in storage costs has led to a growing volume of information that is stored in file systems.

This paper describes Eureka, a file system search engine that employs a "structured" view of the world in order to improve the effectiveness of file searches. Eureka is inspired by research in the Web [2, 5] and Hypertext [6, 7] communities, which has shown that the overall *structure* in a collection of hyper-linked documents can play an important role in determining the importance and ranking of different documents. Based on this intuition, we develop a framework for inferring semantic links in a file system, thus transforming a "flat" collection of files in a graph of hyper-linked documents, and quantifying the importance of each file based on the characteristics of this semantic graph. The end goal, of course, is to use this information in order to derive more effective rankings of the search results.

Our proposed approach is in contrast with existing file system search engines, such as, Google Desktop[1] or Spotlight[2], which ignore any relationships among files and rely solely on conventional IR metrics [1] or localized file meta-data for determining result rankings. To the best of our knowledge, ours is the first work to explore the inference of semantic links among a collection of files in order to provide a more effective ranking of search results.

2. OVERVIEW OF EUREKA

Similar to conventional file system search engines, Eureka enables the efficient retrieval of files based on a set of *search keywords.* A file belongs in the result of a search if it contains the specified keywords, and the matching files are ranked based on their relevance to the search terms in order to assist the user in locating useful results. The key novelty of Eureka, however, is that it takes into account the relationships among files in order to derive more effective rankings of the search results. To illustrate this idea, consider two source files 'htable.h' and htable.cpp' that implement a specific hash table class 'htable'. Intuitively, there exists a relationship between the two files, as the source file includes the header file; moreover, this relationship can be detected fairly easily as the two files have a similar name, while the contents of 'htable.cpp' reference 'htable.h' by name. As we argue in this paper, this information is important in ranking the two files if they appear together in the result of a search, as their semantic relationships impose a intuitive ordering between the two. In this particular case, for instance, it might be useful to rank the header file higher in a search for 'htable', as the source file intuitively depends on it.

HT'05, September 6–9, 2005, Salzburg, Austria.
Copyright 2005 ACM 1-59593-168-6/05/0009 ...$5.00.

[1] `http://desktop.google.com`
[2] `http://www.apple.com/macosx/features/spotlight`

The previous example illustrates the crux behind the design of the Eureka search engine. Essentially, our proposed system automatically infers semantic relationships among files and transforms the file system in a graph of linked documents. Based on the characteristics of this semantic network, Eureka computes the FileRank of each file, that is, an intuitive metric that quantifies the importance of the file in the information space of the file system. The key idea is that this metric can bias the ranking of search results toward the more "important" files, thus boosting the effectiveness of conventional IR-based rankings.

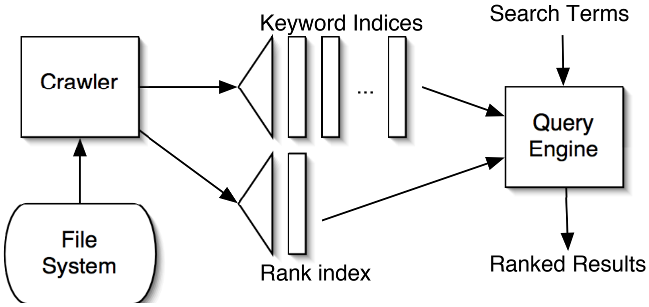

Figure 1: Architecture of Eureka

As shown in Figure 1, Eureka consists of two main components, namely, the *Crawler* and the *Query Engine*. The *Crawler* scans the file system and creates two types of index structures: (a) keyword indices, that record the occurrence of keywords in the contents of files, and (b) a rank index, that stores the FileRank of each file. As we mentioned earlier, the latter is our proposed importance metric for files and is determined from the semantic file graph that is built during the scan. The *Query Engine* uses the keyword indices to compute the set of files containing the search terms, and determines the rank of each result by scaling a conventional IR-based metric [1] (in this case, cosine similarity) with the FileRank of the corresponding file (which is always between 0 and 1).

3. COMPUTING FILE IMPORTANCE

In this section we discuss two key points of our approach: inferring links among files, and deriving the importance of different files in the file system.

3.1 Inferring Semantic Links

We represent a file system as a set of files $\mathcal{F} = \{F_1, \ldots, F_n\}$, where each F_i refers to any stored object, e.g., a directory, a regular file, or a symbolic link. Each F_i is characterized by a set of keywords $\texttt{kwords}(F_i)$ that are extracted from the contents of the file (using type-specific keyword extractor programs) or its meta-data.

We define the concept of the *semantic file graph* G, where each node corresponds to a file and edges encode the semantic relationships among different files. Formally, an edge (F_i, F_j) denotes a semantic relationship between files F_i and F_j and is associated with a weight w_{ij}, indicating the strength of the relationship. Note that two files may be connected with relationships from different classes, and hence might be linked with multiple edges. In our work, we define three types of semantic links, namely, *Content Overlap*, *Name Overlap*, and *Reference* links, that are inferred automatically based on the contents and meta-data of files . In what follows, we discuss briefly the definition of each link and the computation of weights.

– Content Overlap Link. This link attempts to capture associations between files that have a high overlap in their content, the intuition being that F_i, F_j are likely to be related if F_i contains a large part of F_j. A typical example of this case is an incoming e-mail message and its reply, which typically contains parts of the original message. The weight of the link is proportional to the overlap between the contents of F_i and F_j and computed as $w_{ij} = |\texttt{kwords}(F_i) \cap \texttt{kwords}(F_j)|/|\texttt{kwords}(F_j)|$, i.e., the size of the content overlap normalized by the size of the target file. Hence, the weight attains its maximum value of 1 if F_j is completely contained in F_i. Since the computation of the overlap can become computationally expensive, Eureka uses min-hash signatures [3, 4] of the files' contents in order to obtain an accurate estimate of the size of the intersection.

– Name Overlap Link. This link tries to capture associations between files that have similar names, e.g., a TeX file 'report.tex' and the bibliography file 'report.bib'. The weight of a name-overlap link is computed as $w_{ij} = |\texttt{name}(F_i) \cap \texttt{name}(F_j)|/|\texttt{name}(F_j)| \cdot q^{b_{ij}}$, where q is a constant in [0,1], and $b_{ij} = 1$ if F_i has been created later than F_i and 0 otherwise. Hence, a link is stronger if the overlap is high and the source file is newer than the target file, e.g., the weight from 'eureka.bak' to 'eureka.cpp' will be higher than the reverse direction, even though the two names have exactly the same overlap. As we discuss later, this will boost the importance of the older file.

– Name Reference Link. This link captures the reference of a file in the contents of another file. A typical example is a directory and the files it contains, whose names appear in the contents of the directory file. (Another similar example are symbolic links, or file aliases.) The weight of the link is $w_{ij} = 1$ if $\texttt{name}(F_j)$ appears in $\texttt{kwords}(F_i)$ and 0 otherwise. By definition, every directory has a reference link to the files (or directories) that it contains, and hence our semantic graph will always contain the directory hierarchy as a subgraph.

We believe that these three types of links capture a large class of common relationships that exist among files. Moreover, they can be inferred automatically based on the contents and metadata of files. (We note that it is straightforward to extend our framework with additional semantic relationships.) Finally, it is important to observe that the defined semantic links are not symmetrical, i.e., $w_{ij} \neq w_{ji}$ in the general case. As we discuss later, this directionality is an important feature of our framework as it enables our ranking technique to locate "important" files in the semantic graph.

3.2 Filerank Computation

We now discuss the computation of FileRank, our proposed metric for quantifying the importance of different files based on the inferred semantic graph. As we have discussed earlier, FileRank is used by Eureka to scale the IR rankings of search results and bias them toward the more "important" files.

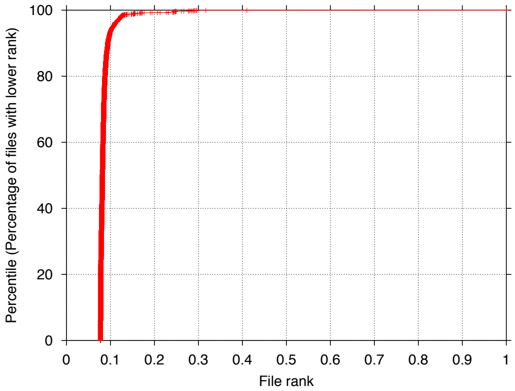

Figure 2: CDF of FileRanks in a real-life data set.

Our approach is inspired by Hypertext [6, 7] and Web-based [2, 5] techniques, where the importance of a document is determined by the number and type of links that reach it. More formally, our technique performs a random walk over the semantic file graph where the probability of traversing a link is proportional to its weight. The FileRank of a file F is simply defined as the probability of visiting F during this random walk, which, intuitively, is higher if the file has neighbors with high ranks and is the target of strong relationships. (This is similar to the well-known PageRank [2] algorithm that is used by Google.)

Figure 2 shows the CDF of (normalized) FileRank values in a sample real-life data set of 5000 ASCII files. (The files represent a random uniform sample from the authors' home directories.) The results indicate that a small subset of files (around 10%) have high FileRanks, thus corroborating our assumption that certain files are likely to be central in the information content of the file system.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we describe Eureka, a file system search engine that integrates Hypertext- and Web-based techniques by adopting a more structured view of file systems. Eureka is based on a simple, yet powerful framework that automatically infers semantic relationships among files and thus transforms a conventional directory-based file system in a network of hyper-linked documents. Based on this model, we propose the FileRank metric for examining the characteristics of the semantic graph and essentially quantifying the importance of different files in the file system. In the same spirit as Internet search-engines, Eureka combines FileRank with conventional IR metrics, in order to derive more effective rankings of search results.

As part of our ongoing work on Eureka, we plan to conduct an experimental study in order to examine more closely the inter-play between FileRank and conventional IR-based rankings. In the same direction, we intend to explore different types of browsing behavior in order to model the traversal of the semantic graph and derive an effective FileRank metric. Finally, an important direction of future work involves the incremental update of the FileRank index, as it is crucial to keep the search engine up-to-date with the current state of the file system.

5. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison Wesley Longman, 1999.

[2] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks*, 30(1–7):107–117, 1998.

[3] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. "Minwise Independent Permutations". In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, May 1998.

[4] Zhiyuan Chen, Flip Korn, Nick Koudas, and S. Muthukrishnan. Selectivity Estimation for Boolean Queries. In *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA*, 2000.

[5] Jon M. Kleinberg. Hubs, authorities, and communities. *ACM Comput. Surv.*, 31(4es), 1999.

[6] Kazunari Sugiyama, Kenji Hatano, Masatoshi Yoshikawa, and Shunsuke Uemura. "Refinement of TF-IDF schemes for web pages using their hyperlinked neighboring pages". In *HYPERTEXT 2003, Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*, pages 198–207, 2003.

[7] Baoyao Zhou, Jinlin Chen, Jin Shi, Hongjiang Zhang, and Qiufeng Wu. "Website link structure evaluation and improvement based on user visiting patterns". In *HYPERTEXT '01: Proceedings of the twelfth ACM conference on Hypertext and Hypermedia*, pages 241–244, 2001.