Mapping Web Pages to Database Records via Link Paths

Tim Weninger Fabio Fumarola† Jiawei Han Donato Malerba†
University of Illinois at Urbana-Champaign
† Università degli Studi di Bari "Aldo Moro"
weninge1@illinois.edu, ffumarola@di.uniba.it, hanj@illinois.edu, malerba@di.uniba.it

ABSTRACT

In this paper we propose a new knowledge management task which aims to map Web pages to their corresponding records in a structured database. For example, the DBLP database contains records for many computer scientists, and most of these persons have public Web pages; if we can map the database record with the appropriate Web page then the new information could be used to further describe the person's database record. To accomplish this goal we employ *link paths* which contain anchor texts from multiple paths through the Web ending at the Web page in question. We hypothesize that the information from these link paths can be used to generate an accurate Web page to database record mapping. Experiments on two large, real world data sets, DBLP and IMDB for the structured data and computer science faculty members' Web pages and official movie homepages for the Web page data, show that our method does provide an accurate mapping. Finally, we conclude by issuing a call for further research on this promising new task.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

mapping, Web, link paths, semi-structured data

1. INTRODUCTION

The World Wide Web contains a wealth of information, and it is rapidly expanding in size and scope. Despite the vast complexities of the Web's landscape, billions of people, even young children, are able to navigate the Web with relative ease. This is partly due to the usefulness of modern Web browsers, search engines and Web design techniques, and partly due to the link-based construction of the Web itself. Arguably, the aspect most fundamental and essential to the ongoing operation of the Web is the existence of hyperlinks. These page-to-page links have shown their ability to tame the Web over and over again, and has transformed an otherwise unwieldy mass of documents into information accessible to the World.

Structured databases of all types have grown alongside the World Wide Web. Recently, efforts have been made to bridge the gap between this structured data and the unstructured data from the Web, but most of these efforts have been one-way. That is, most current work focuses on extracting structured information from one or more Web pages. While this is an important task, if technology could be provided to map specific Web pages to records in a database then the structured and unstructured data could be used to mutually enhance each other in order to address many difficult problems. Therefore, mapping structured database records to Web pages is a specific challenge to the database and information retrieval community.

On the World Wide Web, to supplement the numerical PageRank-type probabilities, edges can be assigned labels according to their associated anchor text. For example, the HTML hyperlink `<a href ="www.yorku.ca/cikm10/">CIKM Conference</a>` can be annotated by the anchor text, 'CIKM Conference'. It is a widely accepted practice for search engines to index an inbound link's anchor text because "anchors often provide more accurate descriptions of Web pages than the pages themselves" [1]. This observation did not start with Google, in fact, the idea of indexing incoming anchor text with the page it refers to was implemented in the World Wide Web Worm in 1994 [4], and since then dozens of studies have looked at various ways to leverage anchor text information.

The main contributions of this paper are as follows: (1) We formulate a new knowledge management task which aims to map Web pages to their corresponding structured database record; (2) We define link paths and show that they are able to represent the referenced Web page more effectively than existing methods; (3) We use link paths to generate mappings from Web pages to their appropriate records in a structured database; and (4) We perform a case study to judge the effectiveness of our approach and demonstrate the implications of this new task.

2. LINK PATHS

Let $G = (V, E)$ denote a given graph, where $V = \{v_1, \ldots, v_n\}$ is the set of vertices and $E = \{e_1, \ldots, e_m\} \subset V \times V$ is the set of directed edges, where each edge e_k can be represented by $\langle v_i, v_j \rangle$. A path $p \in G$ is a sequence of directed edges $p = \langle \langle v_1, v_2 \rangle, \ldots, \langle v_{l-1}, v_l \rangle \rangle$. In this paper we denote $u = v_1$ and $v = v_l$ as the source and destination vertices of path p respectively. Each edge is associated with a value called the edge cost denoted

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

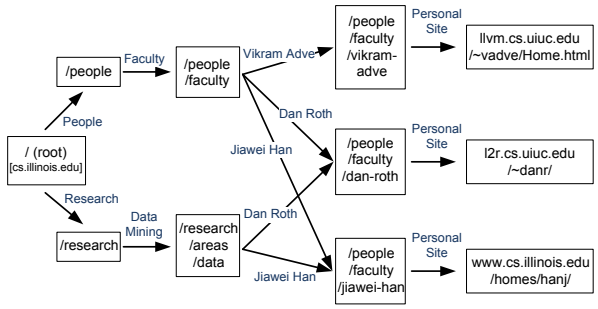

Figure 1: Cropped Web graph from the Computer Science Department at the University of Illinois at Urbana-Champaign

by c_e. For our purposes all edge costs are uniform; so for any path p in G, $c(p) = l$.

On the Web, each Web page represents a vertex and each hyperlink represents a directed edge. A *link path*, therefore, is a path through the Web-graph from one Web page to another. Specifically, if a path between page u and page v contains pages x, and y then a link path from u to v is $u \rightarrow x \rightarrow y \rightarrow v$.

Anchor tags along the link path are extremely important. To capture this information we label each edge in the link path with the corresponding anchor text. If the link between pages u and x has the anchor text a then the link is labeled $u \xrightarrow{a} x$.

Because there are an infinite number of possible paths on the World Wide Web, the first step is to identify the source page u and destination pages $v \in V'$ where $V' \subset V$. The source page u, known as the *reference page*, provides context to the mapping task and is therefore task dependant. For instance, if the task is to map personal Web pages at the University of Illinois to the structured university phone book then an appropriate reference page would be `www.illinois.edu`; if the task is to map official movie Web pages to structured IMDB data then an appropriate reference page may be `http://trailers.apple.com`. In any case, the reference pages should be identified either manually or by some heuristic.

Identifying the set of destination pages V' is similar to the homepage identification task from the 2002 TREC Conference [2] and similar to other work on homepage finding [3, 5]. Otherwise, simple heuristics can be of limited use to extract the destination pages.

Now that the source u and destination V' pages have been identified, we turn our attention to the various link paths between u and each destination page $v \in V'$.

2.1 Finding link paths

A simple way to find the link path between u and v is to perform a shortest path search on the graph. Unfortunately, a single link path may not contain all the information needed to provide an accurate mapping. We propose two solutions to this problem: (1) find the K-shortest link paths, and (2) find the K-shortest loopless [6] link paths. Our intuition is that loopless paths will result in a greater variety of anchor texts than the K-shortest path method because the loopless paths are not susceptible to the cycles within Web site menus. In both the K-shortest and K-shortest loopless path finding methods we would expect to collect a set of anchor texts along various paths between the reference page u and the destination page v.

2.2 Anchor Text

The path p from u to v will have edges $\{e_1, \ldots, e_{l-1}\}$ annotated by the anchor text of each edge a_e. The set of anchor texts $\{a_{e_1}, \ldots, a_{e_{l-1}}\}$ for the path p, denoted A_p, typically contains a descriptive text relative to the reference page u of the destination page v.

Example. The anchor texts retrieved from two paths two Dan Roth's homepage in Figure 1 are: A_{p_1}={*People*, *Faculty*, *Dan Roth*, *Personal Site*} and A_{p_2}={*Research*, *Data Mining*, *Dan Roth*, *Personal Site*}.

From this real world example, we see the utility of anchor texts from link paths because Dan Roth is a *person* and a *faculty* member who does *research* in *data mining*.

Next the K link paths $\{p_1 \ldots p_K\}$ are combined into a bag-of-words representation $\{A_{p_1}, \ldots A_{p_K}\} \in \mathcal{A}_{u,v}$ for each of the K link paths between u and v. We refer to $\mathcal{A}_{u,v}$ as a bag-of-anchors.

Example. The bag-of-anchors from the running example is {*Research:1*, *People:1*, *Faculty:1*, *Data Mining:1*, *Dan Roth:2*, *Personal Site:2*}.

2.3 Aggregating Link Paths

As discussed earlier, there will be many destination pages in each Web site. Therefore, each destination page $v \in V'$ will have its own set of K link paths and its own bag of anchors $\mathcal{A}_{u,v}$ resulting in $|V'|$ bags $\{\mathcal{A}_{u,v_1}, \ldots, \mathcal{A}_{u,v_{|V'|}}\}$.

Example. In the example from Figure 1 there exist three destination pages, and therefore three bags:

$\mathcal{A}_{u,v_1}$={*Research:1*, *People:1*, *Faculty:1*, *Data Mining:1*, *Dan Roth:2*, *Personal Site:2*}

$\mathcal{A}_{u,v_2}$={*Research:1*, *People:1*, *Faculty:1*, *Data Mining:1*, *Jiawei Han:2*, *Personal Site:2*}

$\mathcal{A}_{u,v_3}$={*People:1*, *Faculty:1*, *Vikram Adve:1*, *Personal Site:1*}

The next step is to rank the texts within each bag of anchors so that more descriptive anchor texts are given a higher ranking. To do this, we normalize the score of each word a of the i^{th} bag by

$$\frac{f\left(a \in \mathcal{A}_{u,v_i}\right)}{\sum_{j=1}^{|V'|} f\left(a \in \mathcal{A}_{u,v_j}\right)}$$

where $f(a \in \mathcal{A})$ is the frequency of anchor a in the bag of anchors $\mathcal{A}$. Put more simply, we normalize each word by the frequency each word occurs in an bag of anchors (term frequency) over the frequency of each term in all bags (global term frequency). Finally, we sort the bag of anchors in descending order.

Example. Continuing the example above, the ranked bags of anchors are:

$\mathcal{A}_{u,v_1}$={*Dan Roth*:2/2=1, *Research*:1/2=0.5, *Data Mining*:1/2 =0.5, *Personal Site*:2/5=0.4, *People*:1/3=0.33, *Faculty*:1/3=0.33}

$\mathcal{A}_{u,v_2}$={*Jiawei Han*:2/2=1, *Research*:1/2=0.5, *Data Mining*: 1/2=0.5, *Personal Site*:2/5=0.4, *People*:1/3=0.33, *Faculty*:1/3= 0.33}

$\mathcal{A}_{u,v_2}$={*Vikram Adve*:1/1=1, *People*:1/3=0.33, *Faculty*:1/3= 0.33, *Personal Site*:1/5=0.2}

Thus, we find that the most descriptive terms for each destination page are ranked highest in each list. We especially notice that the anchor text nearest the destination page is not necessarily the most descriptive.

3. MAPPING WEB PAGES TO DATABASE RECORDS

The ranked link path information described in the previous section can be used for many purposes including Web search and information retrieval, and while the IR task may be beneficial it is not

the goal of this work. Instead, our particular goal is to use the texts encoded in the link path to map the destination Web page ($v \in V'$) to its corresponding record in a structured database $r \in R$.

To rephrase, given a set of structured database records, we wish to add a new column to the schema labeled `URL` and populate the new cells of the record with URLs of the corresponding Web pages.

Very frequently the text on a link path is not exactly the same as corresponding text in the database. Names, especially, can be represented in several different ways. For example, a persons name can be represented with or without the middle name, with the middle name abbreviated, last name first, and so on. Therefore, an exact byte-by-byte query would rarely return any results. To mitigate this problem, before a query is actually performed, the anchor text is sanitized, that is, all punctuation an extra spaces are removed and all letters are lowercased.

The actual retrieval function should collect records which match terms from the query string, otherwise the ordering of terms would matter (*e.g.*, 'Dan Roth' would not match 'Roth, Dan'). Most database systems have an indexing or search function to handle these types of queries; we use MySQL and its `match against` function to retrieve records.

3.1 Achieving Strict and Approximate Matching with a Threshold

Frequently the search function will return several candidate matches. To select the optimal match from these results we adapt a word distance heuristic with a threshold. This threshold λ ($0 \leq \lambda \leq 1$) does not allow a mapping to occur when the word distance is above the threshold. For our purposes, we examine the two possible extremes of λ: strict matching ($\lambda = 0$), and approximate matching ($\lambda = 1$).

In strict matching we map a Web page to a database record if and only if an exact match is found, that is, when the word distance is 0. In approximate matching we map a Web page to the database record with the closest word distance.

3.2 Mapping Framework

The overall algorithm, shown in Algorithm 1, should serve as a general framework for the the mapping task.

As input to the framework, we require a set of records (*i.e.*, a structured database), a column from that database to match against, a Web page from which link paths originate called a *reference page*, and a set of Web pages to be mapped to records in the database called *destination pages*. These destination pages most commonly refer to a single entity or item that is dually described in the records of the database. Finding these end pages is the topic of previous and ongoing research known as the homepage finding task.

The output of the framework is a one to one mapping of a record to an destination page. In some problem settings a many to one, one to many or many to many mapping may be preferred, but in this initial work we only consider singular mappings. These mappings can be used for any number of tasks; we describe some of the potential outcomes resulting from accurate record to Web page mappings in Section 6.

4. EXPERIMENTS

We evaluated the effectiveness of our algorithm using two data sets, and we compare the performance of our methods against two worthy baseline systems. Despite the large number of studies related to this work, to the best of our knowledge there do not exist other methods for mapping Web pages to structured records. Therefore, we cannot directly compare our algorithm to other published methods.

Algorithm 1: Mapping Framework

input : Set of records $r \in R$, Selected column $c \in DB$,
Reference page u, Destination pages $v \in V'$
output: One to one mapping M

```
foreach v ∈ V' do
    Find link paths P_K from u → v;   /* Section 2.1 */
    Sorted anchors A_{u,v} from P_K;   /* Section 2.3 */
    foreach anchor text a ∈ A_{u,v} do
        Find record r matching a on column c;
        /* Section 3 */
        if match found then
            add r − v to M;
            break;
return M;
```

Our first data set is a crawl of the departmental Web sites of the top 25 American computer science graduate schools[1]. The second data set is a crawl of movie Web pages starting from `http://trailers.apple.com` and `http://www.allmovie.com`. Given these data sets the task is to map faculty members' homepages to their records in DBLP and movie Web pages to their record in IMDB.

4.1 Baselines

One of the goals of this section is to compare the expressive power of link paths to the current method which uses only adjacent links. Therefore, for the first baseline we query Yahoo's Site Explorer Service – which returns inbound links to the query page – with each destination Web page $v \in V'$, and we retrieve the first 10 results and extract the corresponding anchor text from the referencing Web page. The resulting 10 anchor texts are ordered by their frequency and mapped to the structured database. The results of this baseline will allow for a comparison between adjacent links and link paths.

The second baseline maps the result of a Google query to the corresponding record in the database.

4.2 Setup

For all experiments we set $K = 10$. The link paths were found between the departments' homepage and each faculty members' personal Web page using the K-shortest path and K-shortest loopless path methods. Likewise, the link paths were found between `http://trailers.apple.com` and `http://www.allmovie.com` and each of the movie Web pages also using the K-shortest path and K-shortest loopless path methods. These paths were combined and ranked as described in Section 2 and mapped to appropriate column using the strict and approximate matching methods from Section 3.

5. RESULTS

Table 1 shows the results of the baseline algorithms. As we alluded to earlier, we find that the MRR results from the Google baseline are comparable to (and in some instances better than) the results from the TREC results (now shown). Based on this observation we believe that the precision at 1 is a good baseline for the mapping task.

Table 2 shows the results for the variations of our algorithm. The first four columns show the the results for the strict matching

[1]Rankings from US News 2010

Table 1: Baseline mapping results.

	Google		Adjacent Only
	MRR	Prec.@1	Prec.
DBLP	.7516	71.58	74.19
IMDB	.4240	23.96	38.97

Table 2: Link paths mapping results

	Strict Matching				≈ Matching
	K-short		Loopless		Loopless
	Prec.	Recall	Prec.	Recall	Prec.
DBLP	96.34	52.86	98.63	54.49	94.32
IMDB	99.69	37.70	99.69	41.85	71.86

method on both the K-shortest path and K-shortest loopless path algorithms. These results show that, even under the strict matching conditions, the mappings are very precise but have low recall. Moreover, the overall accuracy under strict conditions confirms our assertion that anchor texts often represent succinct, canonical representations of the referenced Web page.

In terms of recall, our algorithm will always return a result when there is no threshold to limit the string distance so recall statistics are not necessary. For tasks which require a higher level of precision a lower threshold (λ) may be appropriate. Figure 2 shows how the precision and recall of the DBLP data set fluctuate as λ varies between 0 and 1.

The graph in Figure 2 confirms our assertion that lower thresholds will result in high precision and lower recall. We also see that at its worst the recall is relatively low, but the lowest precision score is not too low. For our purposes we prefer $\lambda = 1$ because the small drop in precision is worth the large gain in recall.

6. CONCLUSIONS

In conclusion, this paper proposes a new information management task: the automatic mapping of Web pages to their corresponding records in a structured database. We show that sorted link paths can be used to achieve this mapping, and we perform experiments on two large real-world data sets that show the effectiveness of our algorithm. Finally, we demonstrate that our approach is able to generate an accurate mapping across two large and varied data sets. Furthermore, we believe that our method does generalize to other data sets, and we encourage the community to explore this approach on more data sets.

Linking unstructured data with the records in structured database has become a popular task because researchers have recognized that there is a greater use for information when it is available in a structured or otherwise organized form. Yet most of the current work has revolved around extracting records from a Web page or ranking documents for retrieval. While these are important tasks, we argue that there is a greater need for a framework which employs structured data to enhance unstructured data, and conversely, employs unstructured data to improve the expressiveness of structured data. We believe that the mapping algorithm proposed in this paper is a promising step towards the development such a framework.

6.1 Future Work

The task of mapping Web pages to database records is not an end in itself, but rather an intermediary step to a number of other tasks. For instance, unstructured data (from a Web page) can be used to aid in the retrieval of semantically related database records, or to extend the database schema, etc. Alternatively, structured data can be used in unstructured tasks in order to categorize search results or improve ranking algorithms, etc.

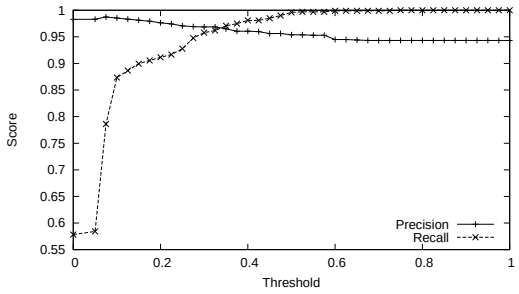

Figure 2: Precision and Recall tradeoff for DBLP data with K-shortest loopless paths as λ varies from 0 to 1.

In terms of the specific link path method, we do not argue that our approach is optimal. There may indeed exist extensions to our method that improve the mapping performance. For example, by modifying the link path weighting algorithm to apply a higher weight to anchor texts closer to the destination page may be more effective than our ranking approach.

Otherwise, the use of title text, URL text, and page content information may be used to improve the overall accuracy of the mapping algorithm. We intend to continue this line of research, and expect to develop a mutually enhancing Web-database system based on this framework in the near future.

7. ACKNOWLEDGMENTS

We would like to thank Jordan Weninger for her help retrieving and labeling data. This work is funded by an NDSEG Fellowship to the first author. The second and fourth authors are supported by both the project "Knowledge Discovery in Relation Domains" funded by the University of Bari "Aldo Moro" and the Strategic Project DIPIS (Distributed Production as Innovative Systems) funded by Apulian Region.

8. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[2] N. Craswell and D. Hawking. Overview of the trec-2002 web track. In *TREC '02: In Proceedings of the eleventh text retrieval conference TREC-2002*, pages 86–95. NIST, 2003.

[3] N. Craswell, D. Hawking, and S. Robertson. Effective site finding using link anchor information. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 250–257, New York, NY, USA, 2001. ACM.

[4] O. A. McBryan. Genvl and wwww: tools for taming the web. In *WWW1: Proceedings of the 15th international conference on World Wide Web*, 1994.

[5] W. Xi, E. A. Fox, R. P. Tan, and J. Shu. Machine learning approach for homepage finding task. In *SPIRE 2002: Proceedings of the 9th International Symposium on String Processing and Information Retrieval*, pages 145–159, London, UK, 2002. Springer-Verlag.

[6] Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(1):712–716, 1971.