The Role of Web Services in Information Search

Jens Graupmann, Gerhard Weikum
University of Saarland, Saarbruecken, Germany
{graupman,weikum}@cs.uni-sb.de

Abstract

State-of-the-art Web search engines are inherently limited in their abilities to search information in Deep Web beyond portals. This paper discusses how Web services and Semantic-Web-style ontologies can be leveraged for a new kind of Deep Web search tool. We argue that keyword based search should be replaced by a more expressive style of concept based querying, and we outline how portals can be automatically wrapped into Web services and how a focused crawl can automatically generate queries for exploring information beyond portal boundaries. As a first, simple demonstrator for the usefulness of the developed framework and tool suite, we briefly present an automatically generated meta portal about movies.

1 Introduction

The Web and its many information sources in Web accessible databases have the potential for being the world's largest knowledge base, encompassing all kinds of business oriented and scientific areas. The predominant way of searching this huge information wealth still is via Google-like Web search engines using keyword based queries. The inherent problems and limitations of this IR-style search paradigm are well known [DE00a], and the general hope is that the next generation of the Web will have technical means for boosting our abilities to search and find information. In particular, Web service technology is expected to improve the organization and accessibility of rich information sources. This paper discusses how Web services can indeed contribute to better search capabilities. More specifically, our approach aims to leverage three major trends that reflect the rapid evolution of the Web as a universal information infrastructure:

- The *Deep Web* (aka. Hidden Web) comprises all information that reside in autonomous databases behind portals, and this data cannot be reached by traditional crawlers [RG01]. While this phenomenon is known for many years, the growing number of Web portals with dynamically generated content and frequently changing hyperlinks aggravate the problem and pose a major challenge.
- The *Automated Web* (or Service Web) addresses the need for streamlining the programmed access to information and e-service portals using WSDL interfaces, as opposed to wrapping form based HTML pages in an ad hoc manner. While this technology is rooted in the area of B2B data exchange, it offers benefits for incorporating Deep Web sources into Web search tools and could eliminate the existing plethora of ad hoc portal wrappers and gateways. E-commerce sites like Amazon and even search engines like Google start offering Web service interfaces; however, standardized and more explicitly typed interfaces of this kind are only a first step towards "mastering" the Deep Web.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

- One problem that Web services do not address is that of semantic diversity. Information on the same subject may exhibit a wide diversity in terminology and structure. The *Semantic Web* [DE02a] aims to overcome these problems by putting emphasis on the adequate organization of Web data sources, using various kinds of ontologies as metadata frameworks for rich modeling and annotation of information. The most important languages for this purpose are RDF and DAML+OIL; another approach would be using carefully chosen XML tags for specific domains. However, despite the growing awareness of the need for semantic annotations, very few real Web sites are following one of these approaches so far.

Our main thesis explained in this paper is that we need to combine technologies from all of the above three trends in order to enable synergies and make significant progress towards better capabilities for Web search. The paper outlines our ongoing work on building a next generation search engine that leverages Web services for Deep Web access and ontologies for coping with semantic diversity.

2 From Keyword-based to Concept-based Search

Keyword search in the form supported by virtually all Web search engines is inherently inadequate for building a search tool that encompasses multiple portals, say for a meta portal on some specific domain. The reason is that the words that appear in a simple keyword list of a user query cannot be easily mapped to the various fields of a portal's query form or the various input parameters of a Web service. An automatic mapping is feasible only in the trivial case when the portal has a single input field that accepts a list of words. Likewise, advanced tools like focused crawlers [Cha02], which aim to automatically classify crawled Web pages based on training data for one or more topics, are inevitably stopped at portal boundaries (unless a portal offers an explicit link collection in addition to the usual query form). For proceeding beyond the portal the focused crawler would have to automatically generate meaningful words for the input parameters of the portal, and this has so far been perceived as too difficult for an automated tool.

As an example for the above problem and a motivation for our approach consider a user who is looking for the movie "The Caine Mutiny" but forgot the exact title and can only phrase the query "Caine" to some search tool. When the search tool attempts to map this query to the input parameters of some portal, it can at best generate the same kind of generic keyword search on the underlying database. A rich movie portal such as IMDB would probably return many movies starring the actor Michael Caine, leading to poor precision of the search result, and the user would have a hard time finding the requested information in this very large result. The irony of this poor outcome lies in the fact that most portals actually offer richer search capabilities with more specific interfaces for "typed" input parameters. In the example, most movie portals can probably distinguish between searching movie titles and actors or other film making crew. If the initial user query would have been something like "title = Caine", the search tool would have had a much better chance to map the query to appropriate input parameters of the portal; likewise, if the query had indeed requested information about Michael Caine then it should have been phrased in the form "actor = Caine".

In the following we refer to the above kind of refined querying as *concept-based* search. A query generally consists of one or more *concept-value pairs*; for example, "actor" denotes a concept and "Caine" is its value that we are interested in. A more advanced example, either on some computer science portal or the entire Web, would be the query "teaching = database systems & research = XML & country = Germany", looking for German researchers that teach database systems and are working on XML. Such query expressions resemble the capabilities of XML query languages to search for both element (or attribute) names and values [TW02]; the key difference is that we are employing this feature to search the Web, not just schematic sources.

Relative to keyword-based search, concept-based queries are more precise in expressing the actual information demand; however, the underlying data is still semantically diverse or even vague (after all, much of the information is in natural language). To cope with this problem we interpret the equality comparison of the form "concept = value" as a *semantic similarity* test: all concept-value pairs whose concept is closely related to the one in the query and whose value is closely related to the value in the query are considered as approximate matches, and a similarity score is computed for a ranked result list. For similarity testing and scoring we use an ontological network, which is essentially a weighted graph that captures semantic relationships like synonyms, hypernyms (i.e., broader terms), hyponyms (i.e., narrower terms), etc. and derives weights from the correlation of words in a large corpus like the Web [TW02].

3 Concept-based Meta Portal Construction

We are currently implementing concept-based Web search facilities in a comprehensive software framework that leverages Web services and ontologies and is particularly geared for Deep Web search. Our software enables the dynamic generation of meta portals on specific subjects (e.g., movies, gene expression data, or handcraft business). To this end we combine the focused crawler BINGO! that we have developed [SMS02, SBG03] with several other components for converting HTML pages into XML, creating Web service entries for HTML forms and effectively wrapping such HTML pages, searching ontologies for semantically related concepts, and automatially generating queries to Web services from the context of the focused crawl and the results of ontology lookups. Our approach proceeds in the following six steps:

- *Content gathering and query interface:* For creating a topic specific meta portal or when the user asks a concept-based query, we initiate a focused crawl. The crawl is driven by sample data that the user has to provide for training the classifier and as initial seeds for the crawl. When the user merely submits a query and does not provide anything else, our system first calls an external Web search engine and prompts the user to provide feedback on the results and thus select appropriate training data. This is a rather exceptional use case; the standard use would be to first create a meta portal by running the crawler and the other components for an extended time period and then submit queries. Queries are evaluated against the indexes that our system maintains about the crawled information, but when the user is not satisfied with the results she can additionally invoke the focused crawl mode with selected, particularly promising Web pages as seeds.
- *Portal discovery:* Whenever the focused crawler discovers a Web page that contains a form it tests, by looking up a UDDI registry (or using another standardized way for discovering Web service such as WSIL), if a page is already registered as a Web service.
- *Portal wrapping:* When a portal candidate is not yet registered as a Web service, a component is invoked that applies heuristic rules for generating a WSDL entry on the fly. Typically, highlighted text next to form fields will become parameter names, and the type of a form field determines the corresponding parameter type (e.g., an enumeration type for fields with a pulldown menu). The WSDL entry (and also a UDDI entry) can either be registered automatically, or can be placed in a candidate queue for later inspection and possible modification by a human.
- *Ontology searching:* For a dynamically generated portal, the parameter names of its WSDL interface are viewed as important concepts and inserted into a locally maintained ontology index. Furthermore, for enumeration types the possible values are likely to be in close semantic relationship to the corresponding concept and are also added to the ontology index. In addition, we search external ontologies like WordNet [Fel98] for newly found concepts and values, extract related words and their relationships, and add them to the ontology index, too. The external ontologies are manually registered with our software, and are themselves wrapped as Web services.
- *Portal query generation:* The focused crawl is driven by either a topic description, given in the form of training data, or an explicit user query. In either case it attempts to explore the data behind portals by automatically generating queries to the portals and indexing the returned result pages. The query generator attempts to match keywords from the training data or concept names from the user query with parameter names of the portal. If this fails, the local ontology index is searched for related terms and the matching

is reattempted. Once appropriate parameters have been identified for portal search, a limited number of parameter values are generated from the values in the query or topic-specific keywords, with preference given to words that have an is-instance-of semantic relationship to the parameter name.

- *Meta portal construction:* All the Web pages that are retrieved either directly on the Web or via automatically generated portal queries are added to the meta portal repository, with the usual kind of text and link indexing. For both static and dynamically generated HTML pages, a component is invoked that converts these pages into XML format, stripping away all the non-interpretable HTML parts (e.g., Javascript code) and attempting to generate meaningful tags (e.g., from table headings or words in highlighted fonts). All this content is automatically classified and ranked relative to the training data and/or user query, and the user can explore the data in an interactive manner.

Figure 1 depicts the main tasks of portal generation described above. It is important to realize that all the above steps heavily rely on heuristics. Our emphasis so far has been on setting up a comprehensive framework for federated Web search and meta portal construction. We are now starting to experiment with the various options that we support, and the next section briefly presents one of these experiments.

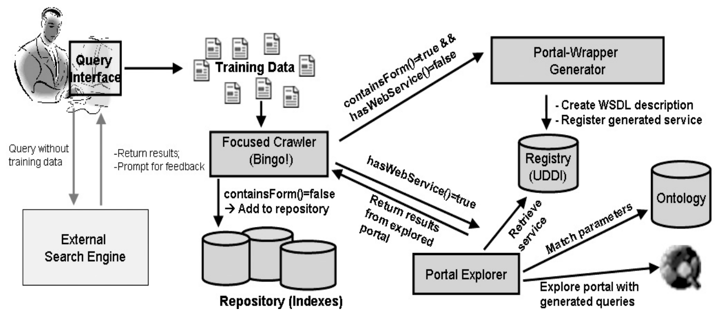

Figure 1: Portal Generation

4 Example Application

We implemented a prototype for an automatically generated meta portal about movies, called MIPS (Movie Information Portal and Search Assistant) that integrates static HTML pages as well as Deep Web sources. The focused crawler BINGO! [SMS02, SBG03] is used for gathering the content. BINGO! analyzes HTML documents (and also other data types like Word, PDF, etc.), computes feature vectors, and automatically classifies documents according to the user-specified topic hierarchy. Part of the analysis is the detection of form fields in the retrieved pages. When potentially relevant forms are found, the Web service Generator is invoked. The Web service Generator automatically creates a wrapper for the form, thus providing a query interface in WSDL. The new service is also added to a private UDDI registry. The system then attempts to classify the Web service into

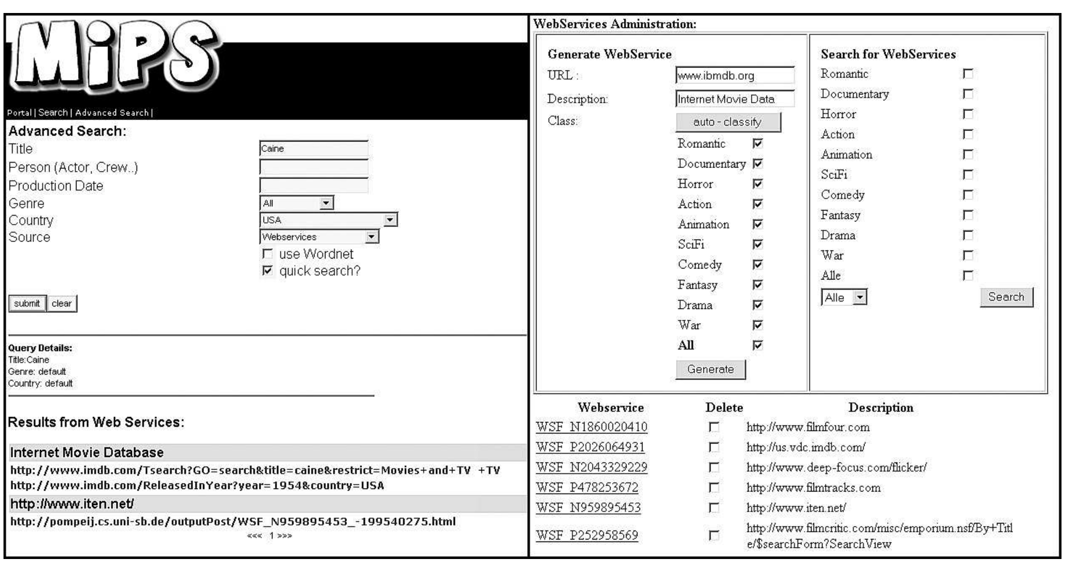

Figure 2: GUI of the MIPS meta portal

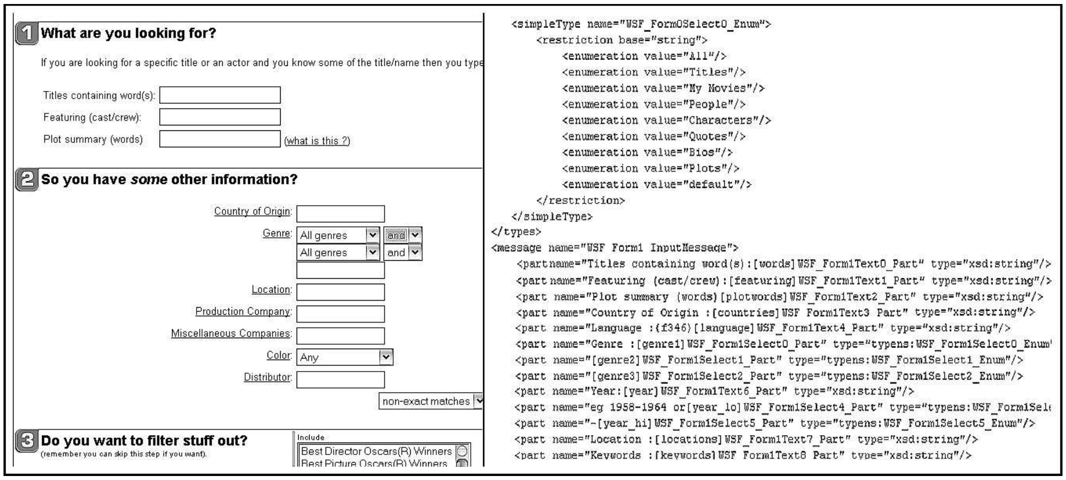

Figure 3: IMDB search form and generated WSDL file

movie genres, based on a list of available genres and small sets of representative movies as training data. On the basis of this information the system generates queries for each Web service and each genre. When the Web service returns results that fit with the training data of the inquired genre, the portal is added to the corresponding topic. If the Web service does not qualify for any genre it is removed from the database.

Figure 2 shows the search form and the Web service administration interface of our prototype. The search form, on the left, shows the result of the previously mentioned example query for “title=Caine”, using only Web services. The administration interface, shown on the right, allows the user to manually add new or delete existing Web services and to invoke the automatic classification. Figure 3 shows an HTML form and the generated WSDL file for the interface of the generated Web service. The names of the method parameters of the generated Web service reflect the descriptors of the form fields and the data types correspond to the valid value assignments of the form fields (e.g. the values of drop down boxes are translated into XML enumeration types). This is important for the invocation of the service, because the arguments of the MIPS search form (Figure 2) are mapped to these parameters. For this purpose the system has its own ontology modeling synonym and hypernym relationships for the movie domain. For performance purposes this mapping is computed in advance and locally stored. For a given query the query processor first retrieves the best matches among the indexed static HTML pages. Then it queries the UDDI registry for matching Web services and invokes them with the precomputed parameter mapping and the appropriate values. Although the returned pages are built dynamically their URL usually includes all information to access the page from its source server. So dynamic pages can now be classified and added to the internal index along with their URLs. For subsequent queries the pages can be searched directly in the index, without calling the Web service.

5 Related Work

Our work has close relationships to several strategic research areas. With the Semantic Web direction we share our interest in ontologies and their role for data integration and schema matching across heterogeneous information sources (see, e.g., [DE02a, DE02b, MBD02]). In the area of Deep Web exploration, the work on wrapper generation (see, e.g., [BFG01, SA99, CGM02]) and on classification of “hidden” databases (see, e.g., [PGS01, RG01]) is highly related to our Web service generation and focused crawling approach. Web services and XML technologies for searching Web and intranet information are also investigated, for example, in [ABMT02, TW02, PV02] . Finally, in terms of the architecture our approach resembles work on federated service architectures like Object Globe [BKK01].

6 Conclusion and Future Work

Although our work is at a relatively early stage, we are confident that the pursued architecture provides an appropriate framework for next-generation search tools. In particular, the combination of Web services with ontologies and the focused crawler paradigm has a great potential that we have to investigate further in larger-scale applications and experiments. Near-future issues in our ongoing work include better forms of automatically tagging HTML and other weakly structured documents, to generate more meaningful and better searchable XML data. So far, our approach to this problem is based on relatively simple heuristic rules; we are considering the use of more advanced machine learning techniques for this purpose (see, e.g., [BDS01, DDH01]). With more data expressed in richly annotated XML form, we plan to carry over some of our prior work on XML querying with ranking [TW02] to a (Deep) Web setting. In particular, we consider adding similarity joins between heterogeneous, but semantically closely related sources to our repertoire of concept based search capabilities.

References

[ABMT02] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo, R. Weber: Active Xml: Peer-to-Peer Data and Web Services, 28th Conference on Very Large Data Bases (VLDB 2002), 2002

[BDS01] V. Borkar, K. Deshmukh, S. Sarawagi: Automatic Segmentation of Text into Structured Records, ACM SIGMOD Conference (SIGMOD 2001), 2001

[BKK01] R. Braumandl et al.: ObjectGlobe: Ubiquitous Query Processing on the Internet, In: The VLDB Journal: Special Issue on E-Services, Vol. 10, No. 3, 2001

[BFG01] R. Baumgartner, S. Flesca, G. Gottlob: Visual Web Information Extraction with Lixto, 27th Conference on Very Large Data Bases (VLDB 2001), 2001

[Cha02] S. Chakrabarti: Mining the Web: Discovering Knowledge from Hypertext Data, Morgan Kaufmann, 2002

[CGM02] V.Crescenzi, G. Mecca, P. Merialdo: RoadRunner: Automatic Data Extraction from Data-Intensive Web Sites, ACM Sigmod Conference (SIGMOD 2002), 2002

[DDH01] A.Doan, P.Domingos, A. Halevy: Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach, ACM Sigmod Conference (SIGMOD 2001), 2001

[DE00a] Special Issue on Next-Generation Web Search, Data Enginieering Bulletin, Vol.23 No.3, 2000

[DE02a] Special Issue on Organizing and Discovering the Semantic Web, Data Enginieering Bulletin, Vol.25 No.1, 2002

[DE02b] Special Issue on Integration Management, Data Enginieering Bulletin, Vol.25 No.3, 2002

[Fel98] C. Fellbaum (Editor): WordNet: An Electronic Lexical Database, MIT Press, 1998

[MBD02] J. Madhavan, P.A. Bernstein, P. Domingos, A. Halevy. Representing and reasoning about mappings between domain models. In Eighteenth National Conference on Artificial Intelligence (AAAI'2002), 2002

[PGS01] G. Panagiotis, L. Gravano, M. Sahami: Probe, Count and Classify: Categorizing Hidden-Web Databases, ACM Sigmod Conference (SIGMOD 2001), 2001

[PV02] Y. Papakonstantinou, V. Vassalos: Architecture and Implementation of an XQuery-based Information Integration Platform, In: [DE02a], 2002

[RG01] S. Raghavan, H. Garcia-Molina: Crawling the Hidden Web , 27th International Conference on Very Large Data Bases (VLDB 2001), 2001

[SA99] A. Sahuguet, F. Azavant: Building light-weight wrappers for legacy Web data-sources using W4F, 25th Conference on Very Large Data Bases (VLDB 1999), 1999

[SBG03] S. Sizov, M. Biwer, J. Graupmann, S. Siersdorfer, M. Theobald, G. Weikum, P. Zimmer: The BINGO! System for Information Portal Generation and Expert Web Search, First Semiannual Conference on Innovative Data Systems Research (CIDR 2003), 2003

[SMS02] S. Sizov, M. Theobald, S. Siersdorfer, G. Weikum: BINGO!: Bookmark-Induced Gathering of Information, 3rd International Conference on Web Information Systems Engineering (WISE 2002), 2002

[TW02] A. Theobald, G. Weikum: The Index-based XXL Search Engine for Querying XML Data with Relevance Ranking, 8th International Conference on Extending Database Technology (EDBT 2002), 2002