Kosmix: Exploring the Deep Web using Taxonomies and Categorization

Anand Rajaraman
Kosmix Corporation, Mountain View, CA, USA
anand@kosmix.com

Abstract

We introduce topic exploration, a new approach to information discovery on the web that differs significantly from conventional web search. We then explain why the Deep Web, an inhospitable region for web crawlers, is emerging as a significant information resource. Finally, we describe the anatomy of Kosmix, the first general-purpose topic exploration engine to harness the Deep Web. The Kosmix approach to the Deep Web leverages a huge taxonomy of millions of topics and their relationships, and differs significantly from that adopted by web search engines such as Google.

1 Introduction

Web search engines, such as those developed by Google, Yahoo, and Microsoft, excel at finding the needle in a haystack: a single fact, a single definitive web page, or the answer to a specific question. Often, however, the user's objective is not to find a needle in a haystack, but to learn about, explore, or understand a broad topic. For example:

- A person diagnosed with diabetes wants to learn all about this disease. The objective is not just to read the conventional medical wisdom, which is a commodity available at hundreds of websites, but also to learn about the latest medical advances and alternative therapies, evaluate the relative efficacy of different treatment options, and connect with fellow-sufferers at patient support groups.
- A reporter researching a story on Hillary Clinton needs access to her biography, images, videos, news, opinions, voting record as a lawmaker, statements of financial assets, cartoons and other political satire.
- A traveler planning a trip to San Francisco needs to learn about attractions, hotels, restaurants, nightlife, suggested itineraries, what to pack and wear, and local events.

These are just three of numerous use cases where the goal is to explore a topic. Topic exploration today is a laborious and time-consuming task, usually involving several searches on conventional web search engines. The problem in many cases is knowing exactly what to search for; in the diabetes example above, if the diabetes

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

sufferer knows there are patient-support groups, or that there might be alternative therapies they might be interested in, it's not hard to find them through conventional search engines. The bigger issue is that most people exploring a topic don't know what to look for.

Kosmix tackles this problem by creating a *topic page* for any topic. The goal of a topic page is to provide a 360-degree view of a topic. In each of the examples above, the Kosmix topic page includes all the information listed. In addition, the topic page also provides a list of related topics, conveniently grouped together, that suggest further areas to explore. The topic page uses a two-dimensional layout that is more reminiscent of a newspaper or a magazine than a search results page. This is not a coincidence. Magazines and newspapers, like topic pages, have been designed for browsing as opposed to search; the goal is to facilitate serendipitous information discovery.

In order to validate the hypothesis that topic pages are valuable for certain use cases, the Kosmix team in 2007 launched RightHealth, a topic exploration engine for health information. In April 2009, the site drew over 7 million unique visitors and served over 20 million topic pages, making it the number 2 health information website in the US according to Hitwise, an independent measurement agency. Having validated the usefulness of topic pages, the Kosmix team in December 2008 launched kosmix.com, extending the RightHealth model to topic pages across all categories. By April, just 4 months after launch, traffic to the new site has grown to over 3 million unique monthly visitors.

A second distinction between Kosmix and search engines lies in the source of the information on the Kosmix topic page. Search engines construct their search results pages from their indexes, which in turn are built using web crawlers. A Kosmix topic page consists of a collection of modules. Each module is constructed using the result of an API call to a web service, done at the time of page construction. This allows Kosmix to tap into the Deep Web, the portion of the web not accessible to crawlers: social networking sites, media-sharing sites (photos, videos, documents), library catalogs, airline reservation systems, phone books, scientific databases and all kinds of information that lie concealed from view behind web forms. Some estimates have pegged the size of the Deep Web at upto 500 times larger than the Surface Web [13].

Although there are differences in the ranking algorithms used by different search engines, the fundamental architecture of web search engines is well-understood [2]. Of late, there has also been some interest in adding Deep Web data to search engines using a enhanced web crawler [8]. A few domain-specific vertical search engines, such as Mobissimo and Kayak, have adopted a federated search approach to the Deep Web, accessing deep web sources at query-time and constructing results pages based on their responses. This paper describes the anatomy of the Kosmix explore engine, which to our knowledge is the first general-purpose, domain-independent service that uses a hybrid of the two approaches to harness the Deep Web.

The rest of this paper is organized as follows. Section 2 introduces the Deep Web and explains some of the industry trends that make the Deep Web more relevant by the day. Section 3 describes the two approaches to harnessing the power of the Deep Web and evaluates their advantages and disadvantages. Section 4 describes the anatomy of the Kosmix explore engine. Section 5 describes related work. Section 6 concludes with some thoughts on the evolution of the Deep Web.

2 The Deep Web

While the Deep Web has existed for almost as long as the Surface Web, a confluence of industry trends has made it more relevant today than at any time in the past.

User-generated content and social media. The Web 2.0 revolution has seen an explosion of sites dedicated to user-generated content (UGC). Examples include Wikipedia, YouTube, Flickr, and specialized sites in many subject areas such as TripAdvisor and Yelp. Such sites have removed friction from the content-creation process and lead to a sharp increase both in the number of contributors and in content of various media types. While search engines have strived to keep up with the deluge, the amount of information available on such services is growing faster than search engine index sizes. Social media sites such as Facebook, MySpace, and Twitter take this trend even further. The information on such sites is subject to various access controls. For example, users may permit only their "friends" to view certain information. In such cases, web crawlers may not have access to the information, since they don't work on behalf of a real user.

Real-Time. One of the weaknesses of the conventional web search architecture is the time lag introduced by the crawl and index process. While search engines have reduced this latency considerably over the past years, it is still a problem for information of a time-sensitive nature. Examples include ticketing and reservation systems of all kinds, auctions and shopping sites with limited product availability, and financial information related to the stock market.

One example that has captured the popular imagination of late is Twitter. When breaking-news events happen, Twitter users are often the first to post news and images online, often within seconds. Instances of this kind include the earthquake in China (May 2008) and the landing of US Airways 1549 on the Hudson River (February 2009). In such cases, there is a window of time where Twitter Search produces superior results over any conventional web search engine.

Specialized search engines. Search ranking algorithms such as PageRank evolved at a time when the web consisted primarily of hyperlinked HTML documents. However, a large fraction of useful content available today does not fit the old model and requires different ranking methodologies. For example, media-sharing sites such as YouTube and Flickr have access to information such as the reputation of the person who uploaded the content, number of views, and ratings. Another trend is the emergence of specialized search engines for specific tasks, such as Mobissimo and Kayak for travel; SimplyHired for job listings; Shopping.com and TheFind.com for products; and so on. As the web evolves from static documents to dynamic content repositories, it would seem that the most natural approach is to let such sites develop their own site-specific search engines, and then federate the results of these engines, or of domain-specific aggregators.

Information presentation. The results presentation model of web search engines has not evolved significantly in over 10 years. In the meantime, specialized services (either site-specific search engines, or category-specific aggregators) have developed innovative information presentation models for specific use cases. For example, Zillow for home prices; TheFind for products; Facebook for user profiles; and so on. Once again, a federated approach makes a lot of sense. Yahoo's SearchMonkey effort is a first step by search engines to address this issue. It allows content owners to submit a catalog of *rich snippets* for a set of queries. The limitation of this aproach is that the set of queries needs to be specified in advance, together with a static snippet for each query.

Availability of APIs. A growing trend is the evolution of websites into web services. Web services provide access to the contents and capabilities of the site via APIs. A handful of standards have evolved for such APIs, including REST and JSON. Crucially, we have reached a tipping point where the quantity of useful information available through such APIs is sufficient to build useful services.

Business model issues. The dominance of search engines as gateways to the web tends to commoditize web content and intermediate between content creators and consumers. Many content creators are therefore wary of allowing search engines access to their entire data set. For example, Twitter does not provide search engines except Twitter Search access to its API. Facebook allows only limited access to search engines. Several newspapers have digitized archives that date back several decades, but do not make these archives available to search engine crawlers. Record labels do not make copyrighted music available for crawl. Such information can in many cases be accessed only through an API.

Infinite Wine in a Finite Bottle. Content created by algorithms is an interesting new trend.The beginnings of this trend can bee seen in the Wolfram Alpha, which can answer questions that are mathematical computations over data. Since the set of mathematical computations is infinite, trying to represent the Alpha as a finite set of web pages to fit in an index is a futile exercise. An API is the only sensible way to access such sites.

3 Approaches to Deep Web Search and Exploration

There are two fundamentally different approaches to incorporating the deep web into search or topic exploration engines.

- **Deep Web Crawl.** Crawl as much of the deep web as possible and incorporate it into a conventional search engine index.
- **Federated Search.** Use APIs to access deep web sources at query-time and construct results pages based on their responses.

Wright [13], using a fishing analogy, calls these approaches *trawling* and *angling* respectively, while Madhavan et al. [8] calls them *surfacing* and *virtual integration.* These approaches are also analogous to the warehousing and mediation approaches in data integration.

3.1 Deep Web Crawling

The crawl-based approach has the advantage that it fits well with the conventional web search model. The additional documents can be added to the search index and ranked using the existing ranking algorithm.The deep web crawl approach has been employed very effectively at Google, especially for tail queries [8].

The disadvantages of the deep web crawl approach relate to many of the issues described in Section 2. In particular, the deep crawl approach shares many of the shortcomings of the crawl-index-search architecture with respect to indexing social media sites, real-time data, specialized search algorithms and presentations for different data types, business model issues, and the problem of services that can handle infinite query sets.

3.2 Federated Search

The dynamic query approach overcomes many of the limitations of the crawl-index-search approach outlined in Section 2. However, the approach comes with its own set of challenges.

Integrating APIs. Each web service comes with its own API, which uses different parameters from every other API. The results are also formatted differently. A piece of custom code, conventionally called a wrapper, is required to connect to each web service, limiting the scalability of the approach.

Source Selection. Given a topic and thousands of potential information sources, it is not practical to query every information source for each query. For example, it is not meaningful to send the query “diabetes” to TripAdvisor.com, a travel resource. If we indiscriminately query a source for topics irrelevant to it, we run into two issues. The first is the potential to clutter the results page with irrelevant information. The second is overloading web service providers, potentially bringing down their systems or getting them to reject future queries from the explore engine. Unlike in the case of an index-based search engine, source selection needs to be done without having access the content of the source.

Query Transformation. Kosmix users enter free form text queries, while the underlying information sources often support a richer query model. Once we deem an information source as potentially useful for a query, the next task is to rewrite the query in the manner most suited for that source. It should be noted that this problem is very different from that of rewriting structured queries, which has been explored in detail in the context of information integration [11, 7].

Results Layout. As we discussed in Section 2, many data sources present results using innovative methods tailored to the kind of data, and we would like to preserve this richness rather than degenerate to the lowest common denominator. In addition, since results are inherently of different types, it is unnatural to force a linear ranking across them.

Performance. Unlike a web search engine, a federated search engine needs to make external API calls in order to construct its pages. Therefore, we should expect this approach to be inherently slower than web search.

Long response times can, however, have the effect of turning off users. Therefore, it is a huge implementation challenge to keep response times within acceptable limits.

4 The Kosmix Explore Engine

In this section we describe the overall architecture and various design decisions that have gone into the Kosmix Explore Engine. We describe in some detail the Kosmix approach to data source access, source selection, and results layout. Our approach to query rewriting and caching (to tackle the performance challenge described in the previous section) will be described in a companion paper.

4.1 A Hybrid Approach to Data Source Access

In the previous section, we listed some of the pros and cons of the crawl-based and federated search models. In practice, Kosmix uses a hybrid approach that combines features of the crawl and the federated search approaches. Some of the data is indexed locally, while API calls are made in other cases.

One of the modules that is surfaced on every topic page is a web search results module, with results from a leading web search engine (Google). In effect, users get the best of both worlds: the comprehensiveness of conventional web search, with all the other advantages of the federated search model. Since the web search module ensures comprehensiveness, we have chosen to focus only on Deep Web data sources that cannot be handled satisfactorily by the surfacing approach described in [8], which is already incorporated into the web search module. Such data sources fall into a few categories:

- Data sources that cannot be adequately crawled using surfacing techniques, either because of their size (e.g., Flickr), business model reasons (e.g., newspaper archives, music), access permissions (e.g., Twitter, Facebook), or because they are computation engines (e.g., Wolfram Alpha).
- Data sources that use specialized search algorithms (e.g., YouTube, Flickr, Truveo).
- Data sources that present results in unique ways (e.g., TripAdvisor, Yelp).
- Data sources where the data varies rapidly with time (e.g., Twitter, airline ticketing).

Data sources that meet any one of the above criteria are candidates for the federated API approach. At Kosmix we have identified several thousand such services, and have integrated thousands of them into the explore engine. We have developed a tool, called Modulator, that supports the common types of web APIs (e.g., REST, JSON) and makes it a relatively quick task to add a new service. It should be noted that adding a data source to Kosmix is a much simpler task than writing a wrapper for a data integration system, because the explore engine does not need to interpret the results from an API call, which is usually the time-consuming part of creating wrappers.

There is also a class of data sources that are not, strictly speaking, Deep Web sources at all, but have interesting structure that makes it unsatisfactory to view them as just collections of web pages. Examples of such sources include Wikipedia, Chrome (data about various car makes and models), A.D.A.M. (doctor-reviewed summaries of many diseases and conditions), and IMDB (the Internet Movie Database). We have chosen to locally index such databases using a structured data index. Once indexed in this fashion, these structured indexes can be treated as identical to external APIs, only with much higher performance.

4.2 The Taxonomy

The cornerstone of the Kosmix explore engine is its taxonomy and categorization technology. The Kosmix taxonomy consists of millions of topics organized hierarchically, reflecting is-a relationships. For example, San

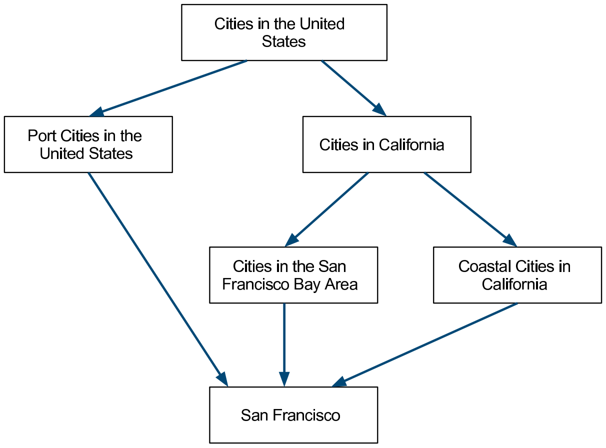
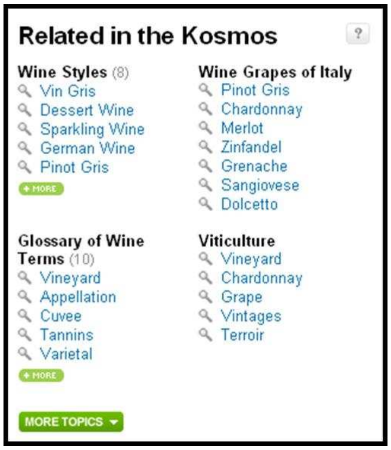

Cities in the United States
Port Cities in the United States
Cities in California
Cities in the San Francisco Bay Area
Coastal Cities in California
San Francisco

Figure 1: A small fragment of the Kosmix taxonomy



Related in the Kosmos

Wine Styles (8)
- Vin Gris
- Dessert Wine
- Sparkling Wine
- German Wine
- Pinot Gris

Wine Grapes of Italy
- Pinot Gris
- Chardonnay
- Merlot
- Zinfandel
- Grenache
- Sangiovese
- Dolcetto

Glossary of Wine Terms (10)
- Vineyard
- Appellation
- Cuvee
- Tannins
- Varietal

Viticulture
- Vineyard
- Chardonnay
- Grape
- Vintages
- Terroir

Figure 2: Topics related to Pinot Noir

Francisco is-a city. The resulting hierarchical structure is a directed acyclic graph (DAG). Figure 1 shows a small piece of the Kosmix taxonomy.

The taxonomy also encodes many relationships beyond is-a. For example, there is a member-of relationship connecting a music group with its members, and a capital-of relationship connecting a country with its capital city. There are many thousands of such relationship types captured in the taxonomy.

The taxonomy has been built over three years using a combination of human curation and algorithmic methods. The raw materials include several publicly available taxonomies, such as DMOZ and Wikipedia, as well as hundreds of special purpose taxonomies in specific fields, such as health, automobiles, and music. The details of how the taxonomy is created and maintained need not concern us here, but the technical challenges we had to surmount include:

- Merging overlapping taxonomies, taking into account that the same concept might be named differently in the two taxonomies.
- Keeping the taxonomy up to date by identifying new topics on an ongoing basis. At Kosmix we gather and analyze millions of RSS feeds every day to identify new topics, such as people, music groups, and so on.
- Adding new relationships between existing topics in the taxonomy.

4.3 Categorization

The second key to the Kosmix explore engine is the Kosmix Categorization Service (KCS). Given a user query, KCS determines the nodes in the taxonomy that are most closely connected with the query. The details of the algorithms involved are proprietary to Kosmix and are not relevant to the discussion here. We will content ourselves with an example illustrating the functionality provided by KCS.

Let us say the query is "Pinot Noir." KCS determines that Pinot Noir is a kind of wine, which is a related to foods and beverages. It also determines that Pinot Noir is a kind of wine grape, and is related to viticulture and vineyards. Figure 2 shows a small selection of the full list of topics KCS determines are related to Pinot Noir. These topics are displayed on the topic page in the section titled *Related in the Kosmos*.

4.4 Source Selection

When a data source is added to the explore engine, it is also associated with nodes in the taxonomy. For example, the TripAdvisor data source is associated with the Hotels node, while Yelp is associated with Restaurants, and WebMD with Health. The Modulator tool mentioned earlier has algorithms that help a human make these associations.

At query time, the explore engine first invokes KCS to determine taxonomy nodes that are related to the query. It then ranks data sources based on how closely they are associated with the query in the topic-space. As one example of a ranking function, we can imagine both the query and data sources as vectors in the topic space, and use a cosine distance measure. In practice, the ranking function used by the explore engine is much more sophisticated, taking into account is-a and other relationships in the taxonomy.

Continuing with the Pinot Noir example, sites deemed relevant to this topic include Epicurious and Food Network (food and recipe-related websites), DailyPlate and FatSecret (both databases listing nutritional value of food), and Amazon.com (for wine shopping). In addition, there are also many general-purpose services that have content on a wide variety of topics, including Wikipedia, Google Image Search, and YouTube.

Once the explore engine identifies a number of candidate data sources (say 15-20) relevant to the topic, it sends the user query to each source. An important caveat is that while the data source may be identified as a candidate for a query, it is not guaranteed to have good results for that query. Therefore, the results from each candidate source are post-processed and then a second ranking function based on the actual returned results is used to determine the final source ranking.

4.5 Laying out the Topic Page

In Section 4.4 we described how the explore engine computes a linear ranking of data sources for a query. The results, however, are not laid out in a linear fashion, but grouped together by information type into a 2-dimensional layout: text, audio, video, user profiles, shopping, conversations and so on. The page real estate allotted to a group varies based on various factors, such as the relevance scores of the data sources in the group. Within a group, each data source gets a variable amount of real estate depending upon its relevance and other factors.

The *Related in the Kosmos* module enables exploration by surfacing topics in the taxonomy that are related to the query, grouped in a fashion that makes it easy to to scan. Figure 2 shows some of the related topics for the query “Pinot Noir.” Clicking on one of these links takes the user to the page for that topic.

5 Related Work

Broder [3] and several other works have classified search queries into three categories: *navigational*, *informational*, and *transactional*. While the exact proportions have varied, there is agreement that a large fraction of queries are informational. Topic exploration is a good metaphor for informational queries.

There is a significant body of work on creating vertical search engines for specific domains by constructing semantic mappings from a mediated schema (or form) to collections of forms within a domain [4, 7, 12, 15]. Most of this work assumes that the mediated schema can be created by hand. Our work, on the other hand, focuses on a general-purpose, domain-independent explore engine. The sheer breadth of queries we need to handle makes it impossible to manually create a mediated schema, leading us to the automated creation and maintenance of a taxonomy.

Google’s approach to crawling the Deep Web is described in [8]. There has been prior work around acquiring documents from databases with restricted query interfaces, as well as computing keyword distributions that summarize database contents to facilitate source selection [1, 5, 6, 9, 10, 14]. There has been significant research on query transformation and rewriting in the context of data integration [7, 11].

6 Conclusion

We described the anatomy of Kosmix, the first general purpose Deep Web Explore Engine. Kosmix enables a new approach to information finding, called topic exploration. We illustrated several use cases satisfied by this model that are not served satisfactorily by today's web search engines. Kosmix uses a hybrid approach to the Deep Web that combines elements of the crawl and federated search approaches. Traffic to Kosmix and its specialized health property RightHealth continue to increase at a rapid clip, demonstrating the value of the Kosmix approach to deep web exploration.

The architecture of web search engines today reflects the historical dominance of the Surface Web. The increasing importance of the Deep Web is forcing a rethink of this architecture. The hybrid approach to the Deep Web has several advantages over both the crawl-only and pure federated approaches. Some standardization around web service APIs could dramatically increase the scalability of this approach. For example, there could be some basic standards around names of input fields such as location-related inputs, text search inputs, category inputs and so on. Even this small amount of standardization can take Deep Web exploration to the next level.

Acknowledgements

The Kosmix explore engine described in this paper is the collective effort of the Kosmix team.

References

[1] L. Barbosa and J. Freire. Siphoning hidden-web data through keyword-based interfaces. In *SBBD*, 2004.

[2] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *WWW7*, 1998.

[3] A. Broder. A taxonomy of web search. In *SIGIR Forum*, 36(2):3-10, 2002.

[4] A. Doan, P. Domingos, A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD*, 2001.

[5] L. Gravano, P. G. Iperiotis, M. Sahami. QProber: A system for automatic classification of deep-web databases. *ACM Transactions on Information Systems*, 21(1):1-41, 2003.

[6] P. G. Iperiotis and L. Gravano. Distributed Search over the Hidden Web: Hierarchical Database Sampling and Selection. In *VLDB*, 2002.

[7] A.Y. Levy, A. Rajaraman, J.J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In *VLDB*, 1996.

[8] J. Madhavan, D. Ko, L. Kot, V. Ganapathy, A. Rasmussen, A. Y. Halevy. Google's Deep Web crawl. *PVLDB* 1(2): 1241-1252, 2008.

[9] A. Ntoulas, P. Zerfos, and J. Cho. Downloading textual hidden-web content through keyword queries. In *JCDL*, 2005.

[10] S. Raghavan and H. Garcia-Molina. Crawling the Hidden Web. In *VLDB*, 2001.

[11] A. Rajaraman, Y. Sagiv, and J.D. Ullman. Answering Queries using Templates with Binding Patterns. In *PODS*, 1995.

[12] J. Wang, J.-R. Wen, F. Lochovsky, and W.-Y. Ma. Instance-based schema matching for web databases by domain-specific query probing. In *VLDB*, 2004.

[13] A. Wright. Searching the Deep Web. In *CACM*, 51(10):14-15, October 2008.

[14] P. Wu, J.-R. Wen, H. Liu, and W.-Y. Ma. Query selection techniques for efficient crawling of structured web sources. In *ICDE*, 2006.

[15] W. Wu, C. Yu, A. Doan, and W. Meng. An interactive clustering-based approach to integrating source query interfaces on the Deep Web. In *SIGMOD*, 2004.