Query Portals: Dynamically Generating Portals for Web Search Queries

Sanjay Agrawal, Kaushik Chakrabarti, Surajit Chaudhuri, Venkatesh Ganti
Arnd Christian König, Dong Xin

Microsoft Research
{sagrawal, kaushik, surajitc, vganti, chrisko, dongxin}@microsoft.com

1. INTRODUCTION

Many informational web search queries seek information about named entities residing in structured databases. The information about entities that a user is seeking could be available from the structured database, or from other web pages. For example, consider an informational query such as [lowlight digital cameras]. A large structured database consisting of products may contain several products that are relevant for the user's query. Surfacing a set of the most relevant products to the user and then enabling her to obtain more information about one or more of them would be very useful. Ideally, we would like a *portal- like* functionality which provides an *overview* of all relevant products, and further allows *drill down* on them.

The need for structured data search is illustrated by the proliferation of vertical search engines for products [3, 1], celebrities [2], etc. Current web search engines already federate queries to one or more structured databases containing information about named entities products, people, movies and locations. Each structured database is searched individually and the relevant structured data items are returned to the web search engine. The search engine gathers the structured search results and displays them along side the web search results. However, this approach does not enable a portal-like functionality due to two key limitations.

Incomplete Results: Current federated search over each structured database is *"silo-ed"* in that it exclusively uses the information in the structured database to find matching entities. That is, the query keywords are matched *only* against the information in the structured database. The results from the structured database search are therefore independent of the results from web search. We refer to this type of structured data search as *silo-ed search*.

While the silo-ed search works well for some queries, it would return *incomplete or even empty results* for a broad class of queries. Consider the query [lowlight digital cameras] against a product database containing the name, description, and technical specifications for each product. `Canon EOS Digital Rebel Xti` may be a relevant product but the query keyword {lowlight} may not occur in its name, description or technical specifications. Silo-ed search over the above product database would fail to return this relevant product. Reviews of the product may describe it using those keywords and can help deduce that the product is relevant to the query. However, the structured database may not contain the comprehensive set of reviews of each product necessary to identify the relevant products. Hence, a silo-ed search against the structured database may miss very relevant results [5]. In fact, current search engines which adopt such a silo-ed search approach over product databases do not return any product entity for our example query.

Inadequate Information for Drill Down: When entities in a structured database are found to be relevant for a user's query, current approaches return *only* information about the entity that is available within the database. The information in the database might be inadequate. Often, the user's need might be better served by information on the web. For example, consider the product `Canon EOS Digital Rebel Xti`. The database might have information about the technical specifications, price, and may be the manual for this product. However, a user might also be interested in reviews, device drivers which are available on the web. Providing links to that information would satisfy her information requirement.

In this paper, we propose the *Query Portals* technology to address the above two limitations by (i) *leveraging web search results* to identify entities in structured databases relevant for informational queries, and (ii) enabling users to *drill down* and obtain specific information from the web on any of these entities. Thus, we attempt to create a dynamic *portal-like* functionality by providing an overview with a set of entities relevant to a web search query, and then allowing users to drill down into one or more of these entities.

We now briefly discuss the intuition behind our approach for addressing the two limitations discussed earlier.

Addressing the Limitation of Incomplete Results: Our main insight for addressing this limitation is to leverage web search results [5]. Our approach is to first establish the *relationships* between web documents and the entities in structured databases. Subsequently, we leverage the top web search results and the relationships between the result documents and the entities to identify the most relevant entities. Consider our example query [lowlight digital cameras]. Several web documents from product review sites, blogs and discussion forums may mention the relevant products in the context of the query keywords {lowlight, digital, cameras}. Therefore, the documents returned by a web search engine are also likely to mention products that are relevant to the user query. We identify the relevant products using the documents returned by a web search engine. Since we leverage web search results, our approach can return entity results for a much wider range of queries compared to silo-ed search [5].

A screenshot of the set of relevant product entities, e.g.

WWW2009, April 20-24, 2009, Madrid, Spain.
.

Figure 1: Related product entities

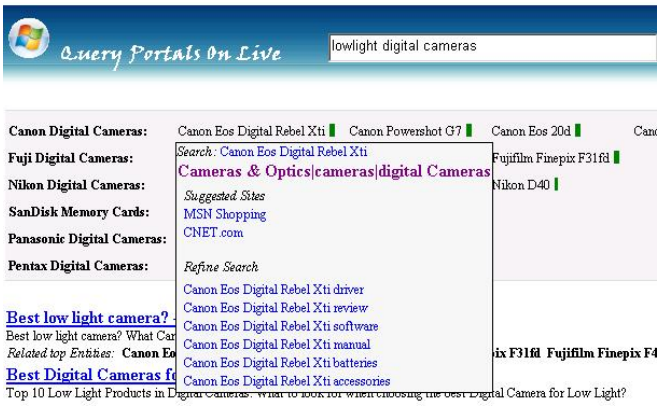

Figure 2: Information links for an entity

`Canon EOS Digital Rebel Xti`, returned by our Query Portals system for the query [lowlight digital cameras] is shown in Figure 1. The set of relevant entities (grouped by the brand name—Canon, Fuji, Nikon, etc.—in this particular example) provides the user with a quick overview of the product entities related to her query.

Overcoming Information Inadequacy for Drill Down: After looking at the set of all relevant entities, the user may want to obtain more entity-specific information, which may not be available in the structured database. Our approach for addressing this limitation is to exploit the content on the web and the web search engines. Specifically, we consider two ways of enabling users to obtain more information on the web about an entity. First, when available, we suggest *authoritative* web sites where a user can find extensive information about an entity. Second, we surface *focused* web search queries per entity to enable a user obtain very specific information on the web about an entity. We refer to the union of authoritative web sites and focused web search queries for a specific entity as *entity information links*. We rely on the query logs, category information about entities, and the web document snapshot in order to automatically identify information links per entity.

The entity information links we show for the example entity `Canon EOS Digital Rebel Xti` is illustrated in Figure 2. We suggest authoritative web sites such as `CNet.com`, or comparison shopping sites such as `MSN Shopping`. We also suggest focused web search queries (such as [Canon EOS Digital Rebel Xti reviews]) to enable a user obtain reviews, batteries, accessories, drivers, RAM memory for this product. Depending on the user's information requirement, she may choose one or more of these suggestions. Note that our approach is complementary to the information about an entity already available in a structured database.

In summary, the Query portals system presents a user with an overview of the entities (along with web search results) relevant to her query and further enables her to obtain specific information about any of the entities.

2. ARCHITECTURE

We now describe the architecture of the Query Portals system. As shown in Figure 3, the system has pre-processing and query-time processing components. The pre-processing components prepare the Query Portals system so that while processing a web search query, the query-time processing components can efficiently identify relevant entities and information links per entity.

The query portals system has two pre-processing components: *entity extraction* and *entity information links identification* components. We first briefly discuss these two components.

Entity Extraction: The first *entity extraction* component takes a collection of web documents as input along with the entity database and outputs a relation consisting for each URL in the web snapshot mentions of entities in the given entity database. The entity extraction component analyzes each web document in the collection and outputs a list of [url, entity name, id, position in document] tuples. We refer to this relation as the *URL-EntityList* relation.

In general, our system can build upon any entity extraction technology [6]. We now sketch the approach adopted in the Query Portals system. Our observation is that the entity database defines the universe of entities we need to extract from web documents. For any entity not in the entity database, we cannot provide the rich experience to a user as the entity is not associated with any additional information. Such entities would therefore not be returned as results. We therefore constrain the set of entities that need to be identified in a web document to be from the given entity database. By doing so, we can avoid the additional effort and time spent by current entity extraction techniques to extract entities not in the target entity database. We leverage this entity database membership constraint to develop techniques (i) which can handle a *wide variety of structured data domains*, and (ii) which are also significantly more *efficient* than traditional entity extraction techniques. The task now is to analyze document sub-strings which match with an entity name in the given database [4].

However, in most realistic scenarios, say for extracting product names, expecting that a sub-string in a web document matches exactly with an entry in a structured database table is very limiting. For example, consider the product entity `Canon EOS Digital Rebel Xti`. In many documents, users may just refer to this product by writing `Canon XTI`. Insisting that sub-strings in documents match exactly with entity names in the reference table may likely cause these product mentions to be not extracted. Therefore, it is very important to consider *approximate matches* between document sub-strings and entity names in a reference table [10, 8]. The approximate matching technology we use in the Query Portals system is described in [9, 8].

Another issue is that of pruning out document sub-strings which match an entity name in the database without intending to refer to the entity. For example, the distinction between the movie "60 seconds" versus a phrase "60 seconds" (in reference to time) is important while extracting a set of movies. We apply known techniques for the entity recognition step [10, 11].

Entity Information Links Identification: The second component identifies authoritative web sites and focused web search queries for each entity in the given structured database. The output of this component is the *Entity-InformationLinks* relation. The information links for an entity consists of a set of authoritative web sites which provide detailed information for the entity, and a set of focused web search queries which may obtain informative web page results about more specific aspects of the entity.

We rely on entity attribute and category information, domain knowledge, query logs, and web document snapshot to identify information links per entity. Due to space constraints, we skip details of the specific techniques. The basic intuition behind our techniques is as follows. If for a number of entities within a certain category, many users are issuing queries of the form [e X], then we hypothesize that for every entity e in the category X is important. In this paper, we focus on X being either a web site domain (such as `CNet` or `MSN shopping`) or a keyword (such as `batteries` or `software manuals`). We also apply other processing (such as stop word removal, stemming, removing approximate duplicates, and a few domain-specific filters) over the web site domains and keywords to ensure that the resulting suggestions are robust and accurate. We use the web site domains as authoritative web sites and the keywords to generate focused web search queries for all entities in the category.

We now briefly discuss the three query-time components.

Entity Retrieval: The task of entity retrieval is to lookup the URL-EntityList relation (materialized by the Entity Extraction pre-processing component) for each of the URLs in the top results from a web search engine for the user's query, and retrieve the set of entities mentioned in the document along with their positions. To enable efficient retrieval of entities per URL, we store the URL-EntityList relation in a database and index it on the URL column.

Entity Aggregation and Ranking: This component ranks the set of all entities returned by the entity retrieval component. We rely on a custom scoring function which takes into account several features for each entity: the number of times the entity is mentioned, the ranks assigned by the web search engine to the documents mentioning the entity, the positions of the entity mentions within the document, the category to which the entity belongs. We only select the entities whose scores are above a pre-determined threshold, and rank them in the descending order of their scores. In general, other scoring functions or rankers based on machine learning techniques may be used in this context, provided we have the required training data [5, 7]. We will investigate such alternative techniques in future.

Information Link Retrieval: The task of this component is to lookup the Entity-InformationLinks relation (materialized by the Entity Information Links Identification preprocessing component) to retrieve the information links for each relevant entity. To enable efficient retrieval of the information links per entity, we store the Entity-InformationLinks relation in a database and index it on the entity id column.

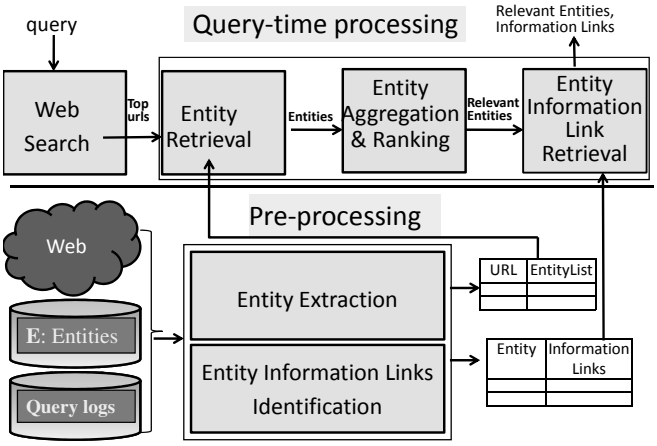

Figure 3: Query Portals Architecture

We then display the entities by grouping them on entity type (people or products) and an interesting attribute of the entity, say, the brand name as shown in Figure 1. Currently, the attributes on which we group relevant entities by are determined upfront per entity category.

3. SUMMARY

The Query Portals system dynamically creates a portal like functionality by creating an overview of all entities relevant to a given query, and then enabling users to drill down and obtain information from the web on specific aspects of an entity. We address the two key limitations of current vertical search engines which *only* search and surface information about entities in a structured database. We address these limitations by establishing the connections between web documents and entities in the given database. We effectively leverage these connections along with a web search engine to achieve the portal like functionality.

4. REFERENCES

[1] http://search.live.com/products/.

[2] http://search.live.com/xrank?form=xrank1.

[3] http://www.google.com/products.

[4] S. Agrawal, K. Chakrabarti, S. Chaudhuri, and V. Ganti. Scalable ad-hoc entity extraction from text collections. In *VLDB*, 2008.

[5] S. Agrawal, K. Chakrabarti, S. Chaudhuri, V. Ganti, C. König, and D. Xin. Exploiting web search engines to search structured databases. In *WWW Conference*, 2009.

[6] D. E. Appelt and D. Israel. Introduction to Information Extraction Technology. *IJCAI-99 Tutorial*, 1999.

[7] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to Rank using Gradient Descent. In *ICML*, 2005.

[8] K. Chakrabarti, S. Chaudhuri, V. Ganti, and D. Xin. An efficient filter for approximate membership checking. In *ACM SIGMOD Conference*, 2008.

[9] S. Chaudhuri, V. Ganti, and D. Xin. Exploiting web search to generate synonyms for entities. In *WWW Conference*, 2009.

[10] W. Cohen and S. Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *ACM SIGKDD Conference*, 2004.

[11] V. Ganti, A. C. König, and R. Vernica. Entity Categorization over Large Document Collections. In *ACM SIGKDD*, 2008.