Removing the MisMatch Headache in XML Keyword Search

Yong Zeng, Zhifeng Bao, Tok Wang Ling
National University of Singapore
zengyong,baozhife,lingtw@comp.nus.edu.sg

Guoliang Li
Tsinghua University
liguoliang@tsinghua.edu.cn

ABSTRACT

In this demo, we study one category of query refinement problems in the context of XML keyword search, where what users search for do not exist in the data while useless results are returned by the search engine. It is a hidden but important problem. We refer to it as the MisMatch problem. We propose a practical yet efficient way to detect the MisMatch problem and generate helpful suggestions to users, namely MisMatch detector and suggester. Our approach can be viewed as a post-processing job of query evaluation. An online XML keyword search engine embedding the MisMatch detector and suggester has been built and is available at [1].

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation

Keywords

MisMatch problem; query refinement

1. INTRODUCTION

Keyword search engine, ever since it was born, has shouldered the responsibility to understand a user's search intention and provide her what she really wants. However, to formulate a precise and effective keyword query is never an easy task. *Query refinement*, which is to give suggestions to help a user revise the query in case that she cannot find what she wants, is a demanding task in building an effective search engine. However, query refinement is a broad and hard topic, and to the best of our knowledge there has never been a systematic categorization on the reasons that trigger query refinement. Therefore, we first briefly present our insight into the categorization, where interested readers can refer to [3] for details.

Category 1: user's query matches her search intention, but what she searches for does not exist in the data. In this case, especially for web search, it is very challenging to know whether what a user searches for does exist or not. Because most likely search engines can always find an html document containing all query keywords, and having nonempty results may easily cover up the fact that what a user searches for does not exist. Although useful suggested queries may still be mined from (sufficiently good) query log data, a user has to read through the results one by one to get a possible conclusion that what she searches for may not exist. Therefore, besides giving a list of suggested queries, it is equally crucial to explicitly tell the user what she searches for does not exist.

Category 2: user's query matches her search intention, and what she searches for also exists in the data. One major reason for problems of this category is: user's domain knowledge is not consistent with the context of the database. E.g., a user issues the query 'cat cancer' while the databases use the phrase 'feline cancer'.

Category 3: user's keyword query does not match her search intention. Typos and underspecified query are the major sources of the problem in this case.

The last two categories are both long standing problems in IR, while Category 1 is a challenge. Furthermore, the structure information exclusive to the (semi-)structured database calls for a structure-awared approach totally different from web search. Therefore, we focus on solving Category 1 on semi-structured data, referred to as the **MisMatch problem**. Let us see a motivation example first.

Example 1. *An XML tree in Fig. 1 describes an online shopping mall. Suppose a user wants to buy a laptop, she prefers Sony's Vaio W and red color, and wants to know its price. Then she may issue a query $Q = \{$'Vaio', 'W', 'red', 'price'$\}$ to search for a laptop. Unfortunately, Vaio W only has three colors: white, blue and pink. However, for existing keyword search methods, they still can find some results containing all query keywords. One of the possible query results by LCA (Lowest Common Ancestor) is the subtree rooted at shop:0.0.0, where keyword 'red' matches one laptop while the rest keywords match another laptop. Obviously, the subtree rooted at* shop *contains too much irrelevant information, i.e. all laptops, which is not expected by the user. In this case, query refinement is needed because what the user wants to search for does not exist.* □

As we can see in Example 1, having nonempty results may cover up the fact that what a user searches for may not exist; and neither existing keyword query refinement works nor the approach of finding both full and partial match results can

SIGIR'13, July 28–August 1, 2013, Dublin, Ireland.
ACM 978-1-4503-2034-4/13/07.

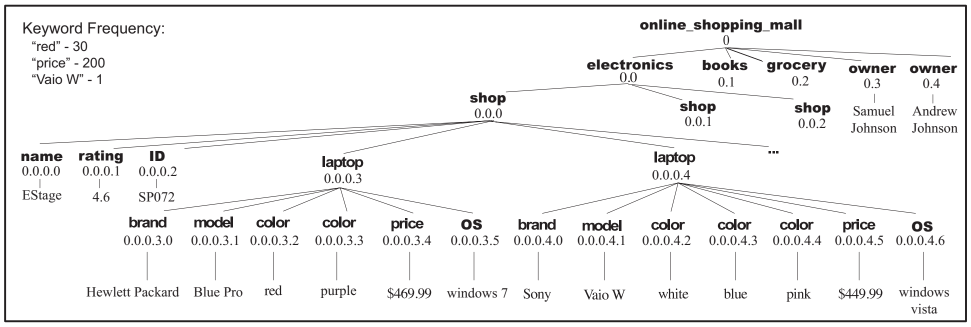
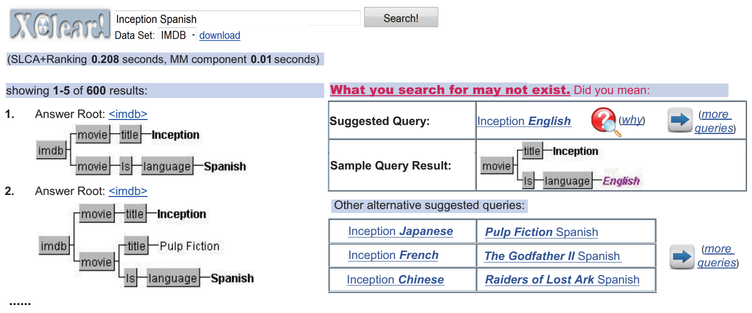


Figure 1: Sample XML Document about an Online Shopping Mall



Inception Spanish
Data Set: IMDB

(SLCA+Ranking **0.208** seconds, MM component **0.01** seconds)

showing **1-5** of **600** results:

1. Answer Root: <imdb>
2. Answer Root: <imdb>

......

What you search for may not exist. Did you mean:

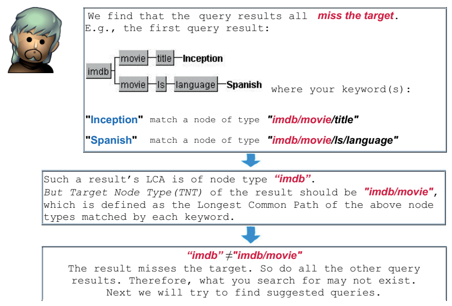
Suggested Query:	Inception ***English*** (why) (more queries)
Sample Query Result:	

Other alternative suggested queries:

Inception ***Japanese***	***Pulp Fiction*** Spanish
Inception ***French***	***The Godfather II*** Spanish
Inception ***Chinese***	***Raiders of Lost Ark*** Spanish

(more queries)

Figure 2: Suggested Queries & Sample Query Result



We find that the query results all *miss the target*.
E.g., the first query result:

where your keyword(s):

"Inception" match a node of type *"imdb/movie/title"*

"Spanish" match a node of type *"imdb/movie/ls/language"*

Such a result's LCA is of node type *"imdb"*.
But Target Node Type(TNT) of the result should be *"imdb/movie"*, which is defined as the Longest Common Path of the above node types matched by each keyword.

"imdb" ≠ *"imdb/movie"*
The result misses the target. So do all the other query results. Therefore, what you search for may not exist.
Next we will try to find suggested queries.

Figure 3: Part of the Reasoning of "why"

solve the MisMatch problem thoroughly, because they do not detect it.

2. XCLEAR AND TECHNIQUES BEHIND

Addressing the MisMatch problem, we develop an XML keyword search engine called XClear, which can (1) show the user why the mismatch exists and (2) provide result-driven suggested queries to bridge the mismatch gap.

Fig. 2 shows a screenshot for a query Q='Inception Spanish' in order to find the Spanish version of a movie Inception. On the left hand side it shows the query results returned by a widely adopted matching semantics SLCA [2]. As we can see from the results, there is no movie Inception with language Spanish. Therefore, help is needed for the user.

Addressing such a problem, as shown in the right part of Fig. 2: (1) XClear gives a notification "What you search for may not exist" to the user. This is a crucial part to form a complete solution to the MisMatch problem. Because without the notification, users have to struggle with reading throughout the results until realizing what they search for may not exist. (2) It provides the best suggested query and its sample result. (3) A "*why*" button (next to the suggested query) is provided for users to get further *reasoning* on why we generate this suggested query. Part of the reasoning page is shown in Fig. 3.

In Fig. 2, users can also view some other *alternative* suggested queries or even find more suggested queries by clicking the *"more queries"* button. All the suggested queries are derived from the XML data and guaranteed to have reasonable query results. E.g., the movie Inception has four languages in the data: English, Japanese, French and Chinese, which correspond to four of the suggested queries provided on the right of Fig. 2.

Techniques behind. Our solution [3] can be viewed as a *post-processing* of the query evaluation, which consists of two components: *MisMatch Problem Detector* and *Suggested Query Generator*. The components accept a list of *all* results retrieved from search engine as input, and the main idea proceeds as follows. We first infer possible user's search targets for the query Q based on its results $\mathbb{R}$; then the detector investigates each result r in $\mathbb{R}$, to check whether it matches one of the possible search targets. If none of them can match a possible target, we claim that Q has the MisMatch problem. Hereafter, to generate suggested queries, we propose a *tf*idf*-inspired scoring measure to help find 'important' keywords in the original query. Then based on each query result r, we try to find some 'approximate' query results which contain these 'important' query keywords and are structurally consistent with r, while having reasonable replacement for the 'less-important' query keywords. Finally, the suggested queries can be inferred from the approximate results.

For a user query that has the MisMatch problem, the output of our suggester consists of three parts: (1) An explicit notification to the user: "what you want to get does not exist". (2) Some suggested queries, and one sample result for each suggested query. (3) An elaborate step-by-step reasoning w.r.t. how each promising suggested query is found.

3. ACKNOWLEDGEMENTS

This research was carried out at the SeSaMe Centre. It is supported by the Singapore NRF under its IRC@SG Funding Initiative and administered by the IDMPO.

4. REFERENCES

[1] *XClear*. http://xclear.comp.nus.edu.sg.
[2] Y. Xu and Y. Papakonstantinou. Efficient keyword search for smallest lcas in xml databases. In *SIGMOD*, 2005.
[3] Y. Zeng, Z. Bao, G. Li, T. W. Ling, and J. Lu. Breaking out the xml mismatch trap. *CoRR*, 1208.2448, 2012.