Exploratory Keyword Search Querying Over Graphs with Quality Guarantees

Casey Fleck[1], Mohammad Hossein Namaki[2], Xin Zhang[2], Yinghui Wu[2], Lawrence Holder[2]

[1] Coastal Carolina University, Conway, SC 29526

[2] Washington State University, Pullman, WA 99164

Introduction

- Can exploratory keyword search querying be streamlined to provide *users* with a quick search algorithm that minimizes query ambiguity and guarantees the quality of the answers at the same time?
- We formulate keyword query refinement as a multi-objective optimization problem subject to a bounded loss of quality. We propose an algorithm that incrementally refines a given keyword query by exploring the promising neighbors of its answer, without re-evaluating all possible query candidates.
- Using real-world graphs, we experimentally verify the efficiency and effectiveness of our algorithms.

Motivation

- Real-world graphs are heterogeneous and noisy. It is usually hard for end users to formulate exact keyword queries, not to mention the quality guarantees returned by the algorithms.
- Since the issued query is only an approximation to the user's true information need, factors such as ambiguity, vocabulary mismatch, and lack of schema can reduce the quality of search results even with well-established keyword search algorithms.
- This calls for automatic query refinement schemes that aid users to obtain accurate results by reducing the ambiguity of the original query.

Problem Definition

Given an initial data graph, G_0, we want to find $|S|$ copies of a single source shortest path algorithm starting from each keyword candidate. We perform a keyword-search expansion that replaces and adds keywords to make the search results more efficient and specific while limiting the amount of executions of the algorithm through the graph. For example, once a user asks for keywords such as *'apple raspberry pi',* we try to expand it to graphs that contain information from Apple and Raspberry Pi to get a more precise answer while still maintaining quality. (Figure 1)

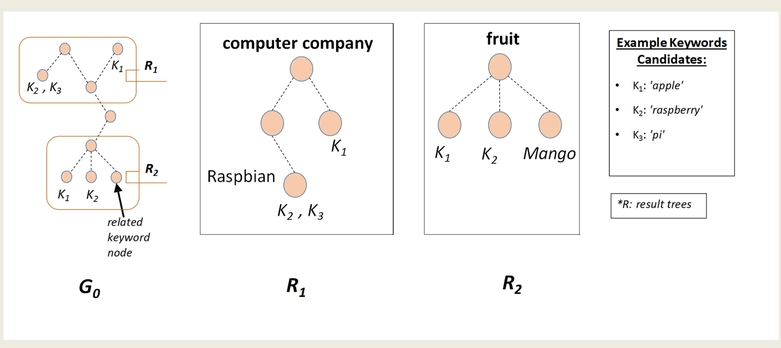

Figure 1: Searching for keywords *'Apple Raspberry Pi'*

Keyword Search (KWS) Refinement Problem with Quality Guarantees

The following is the definition of our algorithm:

- **Input:** Given (1) a data graph, G_0 ; (2) a keyword search query, $Q_i = \{K_1, \dots, K_n\}$; (3) a loss bound $\in \epsilon[0, 1)$; (4) and a distance *a bound b*.
- **Output:** Retrieving the answers of a refined keyword query Q_{i+1} by maximizing the diversity Div.(Q) subject to the quality bound $\in$, where Div.(Q) is defined as the sum of pairwise distance over two keywords and the distance of two keywords K_1, K_2 is defined using *NSWD (Normal Semantic Web Distance)* [ref].

$$\textbf{NSWD}(K_1, K_2) = \frac{\boldsymbol{max}\{\log|C_\lambda(K_1)|, \log|C_\lambda(K_2)|\} - \log|C_\lambda(K_1) \cap C_\lambda(K_2)|}{\log|N| - \min\{\log|C_\lambda(K_1)|, \log|C_\lambda(K_2)|\}}$$

Where $C_\lambda(K_i)$ is type (e.g. label) of K_i neighbors in the data graph.

Intuitively, the same type of nodes are connected to a pair of nodes with similar meaning.

Keyword Search (KWS) Framework

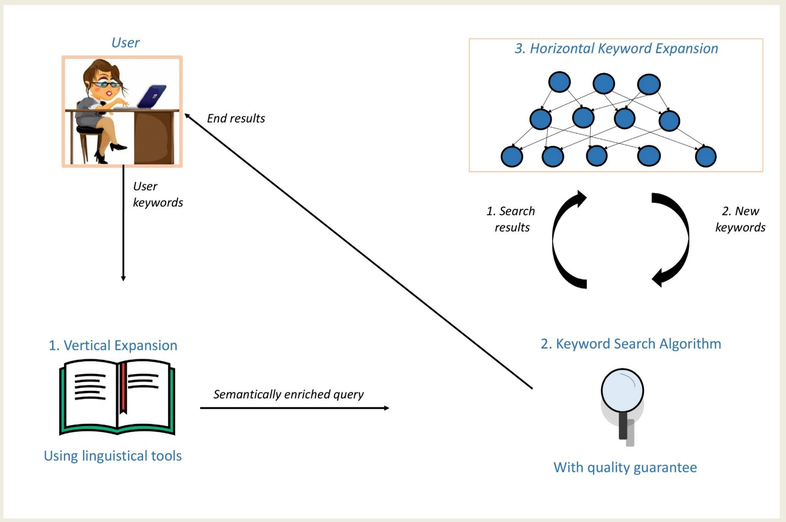

Figure 2: The cycle of exploratory keyword search querying

Highlight Of a Heuristic Algorithm

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis egestas dui non pellentesque euismod. Donec non sem molestie, fringilla leo at, tincidunt mauris. Pellentesque fringilla tellus et feugiat dapibus. Pellentesque rhoncus eu sem at convallis. Nunc orci ex, pellentesque sed imperdiet non, tincidunt vitae sapien. Quisque pharetra est erat, et cursus urna sagittis eget. pellentesque sed imperdiet non, tincidunt vitae sapien. Quisque pharetra est erat, et cursus urna sagittis eget. Integer id tincidunt tortor. Morbi sodales odio eu interdum

Case Study

- Specific knowledge graphs we used our algorithm on.
- Example input query (Q_i) and its answers -> then the refined query suggestion plus the newer results. Explanation of what was found as results
- 2 graphs (figures 3 + 4); Figure 3 showing (maybe a tree?) of the keyword input query and its leave results. Same with the refined query (figure 4).
- Figure 3 (INCLUDE TITLE)
- FIGURE 4 (INCLUDE TITLE)

Performance

- Figure 5: Include table (1) that shows amount of number of searches, amount of diverse search results, and maybe time. Include improvement of efficiency graph (Figure 5) that's linked with table 1.

Conclusion

- Did it limit ambiguity, improve quality, specification, and efficiency of the search according to the tables and graphs?
- How does this show it improves performance/ answers of search results from tables?

Future Work

- How could it be improved, what are others doing out there, what are the next steps in research.

Acknowledgements

This work was supported by the Washington State University Smart Environments program, and the National Science Foundation's Research Experience for Undergraduates program under Grant No. 1460917.