EECS 405 Project Proposal

for

Improving Efficiency of String Similarity Searches through Clustered Pruning

Prepared by

James Fitzpatrick
Kyle Patterson

Version 1.0
March 18, 2014

Revision History

Name	Date	Reasons for Change	Version
James Fitzpatrick	3/18/14	Initial Proposal	

Contents

1. Introduction

String similarity search is a rapidly growing area of research within the computer science community due to its numerous overlapping applications in the modern world. From biology and genetics to computer security, improvements to current string searching techniques must be improved to handle the large amounts of data being gathered today. As presented in [2], pruning techniques were effective in reducing the time of a query to less than 50 milliseconds; however, these pruning techniques were computationally costly, requiring a minimum of 0.1 billion entries in generated triples in the range-based pruning methods. With the move towards big data sets, it may be more beneficial to include common data-mining practices that are already designed for large sets of data.

2. Problem Definition

We propose a method of pruning the search space that utilizes a common data mining technique, clustering. At the application's initialization, we will mine the database and identify similar groups of strings; these clusters will be indexed by longest common substring. When a search is performed at run time, we will search for the appropriate cluster of strings using the constructed cluster index and restrict our search space to the appropriate cluster of data. Although this method includes a significant amount of preprocessing, index construction is only needed when the data is modified and is not present in every search. This effectively reduces the number of strings that must be compared via the B^{ed}-Tree and Top-k techniques from [1] and [2], which should further improve search time among large sets of data.

As presented in [9], using longest common subsequence (LCS) is the second most effective technique in grouping strings (with the Levenshtein distance being the most effective). However, LCS is more conducive to key generation because it provides a testable string that can be used as the key. The best-matching LCS of a query and a key is guaranteed to identify the cluster that contains the query, if the query is present in the database.

3. Goals

1. Research and gain a thorough understanding of the research papers
2. Implement the B^{ed}-Tree and Top-k Algorithms
3. Build clustered datasets based on the Longest Common Subsequence on:
   - Movie Titles from IMDB
   - Publication Titles from DBLP
4. Test the effects on the performance of the proposed clustered search space by comparing to the performance in an unclustered search space
5. Formally present findings through a final presentation

4. Division of Labor

James Fitzpatrick will be implementing the Top-k Algorithim as documented in the papers below. James will also create an agnostic test harness to assess the differences between the different seardh algorithms and the different clustering techniques on the datasets.

Kyle Patterson will be implementing the B^{ed}-Tree Algorithm and will be developing the clustering algorithm for the datasets.

5. Papers Identified

1. Z. Zhang, et. al. "B^{ed}-Tree: An All-Purpose Index Structure for String Similarity Search Based on Edit Disctance", SIGMOD 2010.
2. Don Deng, Guoliang Li, Jianhua Feng, Wen-Syan Li, Top-k String Similarity Search with Edit-Distance Constraints, ICDE 2013.
3. A. Andoni and K. Onak. Approximating edit distance in near-linear time. STOC, pages 199-204, 2009.

4. M. Hadjieleftheriou, A. Chandel, N. Koudas, and D. Srivastava. Fast indexes and algorithms for set similarity selection queries. ICDE 2008, pages 267-276.

5. M. Hadjieleftheriou, N. Koudas, and D. Srivastava. Incremental maintenance of length normalized indexes for approximate string matching. *SIGMOD Conference*, pages 429-440, 2009.

6. W. J. Masek and M. Paterson. A faster algorithm computing string edit distances. *Journal of Computer and System Sciences*, 20(1):18-31, 1980.

7. T. Kahveci and A. K. Singh. Efficient index structures for string databases. VLDB, pages 351-360, 2001.

8. Z. Yang, J. Yu and M. Kitsuregawa. Fast algorithms for top-k approximate string matching. AAAI, 2010.

9. R. Zafarani and H. Liu. Connecting users across social media sites: a behavioral-modeling approach. KDD, pages 41-49, 2013.

10. M. Rafsanjani, Z. Varzaneh, N. Chukanlo. A survey of hierarchical clustering algorithms. TJMCS 5(3):229-240, 2012.