Survey of Spatial Approximate String Search

B.Ramya M.Tech[1]

[1]*Department of Computer Science and Engineering, Karunya University, Coimbatore, Tamil Nadu, India*

Abstract: **Several applications require finding objects closest to a specified location that contains a set of keywords. In view of example, online yellow pages allow users to specify an address and a set of keywords. In response, the user obtains a list of businesses whose description contains these keywords and it ordered according to their distance from the specified address. The complexities of nearest neighbor search on spatial data and keyword search on text data have been extensively studied separately. Still, to the best of our knowledge there is no efficient method to answer spatial keyword queries, exclusively, queries that specify both a location and a set of keywords. We survey the current techniques to cope with the problem of string matching that allows errors. For many fast rising areas such as information retrieval and computational biology this is becoming a more and more relevant issue. We focus on spatial string searching and mostly on edit distance, its statistical behavior, its history and current developments, and the central ideas of the techniques and their difficulties. The aim of this survey is to present an overview of the state of the art in approximate string searching.**

Keywords**: nearest neighbor search, spatial data, keyword search, spatial keyword queries, string matching, edit distance and approximate string searching.**

I. INTRODUCTION

Keyword search over a large amount of data is an important operation in a wide range of fields. Felipe et al. has recently extended its study to spatial databases [17], where keyword search becomes a fundamental building block for an increasing number of real-world applications, and proposed the IR2-Tree. A main limitation of the IR2-Tree is that it only supports exact keyword search. In reality, keyword search for retrieving approximate string matches is required [3], [9], [11], [27], [28], [30], [36], [43]. Because exact match is a special case of approximate string match, it is clear that keyword search by approximate string matches has a much larger pool of purposes.

Approximate string search is necessary when users have a fuzzy search condition, or a spelling error when submitting the query, or the strings in the database contain some degree of uncertainty or error. In the context of spatial databases, approximate string search could be combined with any type of spatial queries. In this survey we focus on range queries and dub such queries as *Spatial Approximate String* (SAS) queries. SAS queries to Euclidean space, depicting a common scenario in location-based services: find all objects within a spatial range r (specified by a rectangular area) that have a description that is similar to “theatre”. We denote SAS queries in Euclidean space as (ESAS) queries. Similarly, SAS queries to road networks (referred as RSAS queries). Given a query point q and a network distance r on a road network, we want to retrieve all objects within distance r to q and with the description similar to “theatre”, where the distance between two points is the length of their shortest path.

Define the similarity between two strings is a key issue in SAS queries. The *edit distance* metric is often adopted [3], [9], [11], [27], [28], [30], [36], [43]. Specifically, given strings $\sigma 1$ and $\sigma 2$, the edit distance between $\sigma 1$ and $\sigma 2$, denoted as $\varepsilon(\sigma 1, \sigma 2)$, is defined as the minimum number of *edit operations* required to transform one string into the other. An insertion, deletion, or substitution of a single character is referred as the *edit operations*. If the different operations have different costs or the costs depend on the characters involved, we speak of *general edit distance*. Or else, if all the operations cost 1, we speak of *simple edit distance* or just *edit distance* (*ed*). In this last case we simply seek for the minimum number of *edit operations* to make both strings equal. For instance *ed* ("survey,""surgery") = 2. The edit distance has received a lot of attention because its generalized version is powerful enough for a wide range of applications. Despite the fact that most existing algorithms concentrate on the simple edit distance, most of them can be easily adapted to the generalized edit distance, and we pay attention to this issue throughout this work.

Another example is Clearly, ε is symmetric, i.e., $\varepsilon(\sigma 1, \sigma 2) = \varepsilon(\sigma 2, \sigma 1)$. For example, let $\sigma 1$ =‘theatre’ and $\sigma 2$ =‘theater’, then $\varepsilon(\sigma 1, \sigma 2)$ = 2, by substituting the first ‘r’ with ‘e’ and the second ‘e’ with ’r’. We *do not* consider the generalized edit distance in which the transposition operator (i.e., swapping two characters in a string while keeping others fixed) is also included. The standard method for computing $\varepsilon(\sigma 1, \sigma 2)$ is a dynamic programming formulation. For two strings with lengths n1 and n2 respectively, it has a complexity of O(n1n2). That said, given the edit distance threshold $\tau = 2$, the answer to the ESAS query is {p4, p7}; the answer to the RSAS query is {p7}. A straightforward solution to any SAS query is to use any existing techniques for answering the spatial component of an SAS query and verify the approximate string match predicate either in post-processing or on the intermediate results of the spatial search.

An important area that we leave aside in this survey is indexed searching, i.e. the process of building a persistent data structure (an index) on the text to speed up the search later. Typical reasons that prevent keeping indices on the text are: extra space requirements (as the indices for approximate searching tend to take many times the text size), volatility of the text (as building the indices is quite costly and needs to be amortized over many searches) and simply inadequacy (as the field of indexed approximate string matching is quite immature and the speedup that the indices provide is not always satisfactory). Indexed approximate searching is a difficult problem, and the area is quite new and active. These issues have been put aside to keep a reasonable scope in the present work. They certainly deserve separate surveys. Goal of this survey is to explain the basic tools of approximate string matching, as many of the extensions

II. RELATED WORKS

There are several studies on answering spatial queries for exact matching of keywords. Chen et al. [1] used separate indices for the spatial and textual information. Zhou et al. [21] suggested a hybrid index that combines spatial and inverted indexes. They use an R*-tree and build inverted indexes for the keywords at the leaf nodes; or use an inverted index to store the keywords and build an R*-tree for each keyword. These techniques do not simultaneously use the spatial and the textual information for pruning. Other studies [14], [5], [4], [20] used the textual and the spatial information conjunctively. They augment a tree-based spatial index with textual information in each node.

The IR2-tree was proposed in [17] to perform exact keyword search with kNN queries in spatial databases. The IR2-tree cannot support spatial approximate string searches, neither their selectivity estimation was addressed therein. Authors in [45], [46] study the m-closest keywords query in Euclidean space, where the proposed bR∗-tree cannot handle the approximate string search neither. Two other relevant studies appear in [7], [13] where ranking queries that combine both the spatial and text relevance to the query object was investigated. Another related work appears in [2] where the LBAK tree was proposed to answer location-based approximate keyword queries which are similar to definition of spatial approximate string queries in the Euclidean space. The basic idea in the LBAK-tree is to augment a tree-based spatial index (such as an R-tree) with q-grams of subtree nodes to support edit-distance based approximate string/keyword searches.

The LBAK-tree was proposed after study on SAS queries in the Euclidean space [44], and it has been compared against the MHR-tree in [2]. Their results have shown that the LBAK-tree has achieved better query time than the MHR-tree, but using more space. Note that the LBAK-tree returns exact answers for the ESAS queries, and the MHR-tree returns approximate answers. The comparison of the LBAK-tree with the MHR-tree said, for ESAS queries, the LBAK-tree should be adopted when exact answers are required; when space consumption must be small and approximate solutions are acceptable, the MHR-tree is the candidate. To the best of our knowledge, RSAS queries and selectivity estimation of SAS queries have not been explored before. Approximate string search alone has been extensively studied in the literature [3], [8], [9], [11], [19], [27], [30], [32], [36], [40], [42], [43]. These works generally assume a similarity function to quantify the closeness between two strings. There are a variety of these functions such as edit distance and Jaccard.

Many approaches leverage the concept of q-grams. The main pruning lemma is based upon a direct extension of q-gram based pruning for edit distance that has been used extensively in the field [19], [40], [42]. Improvements to the q-grams based pruning has also been proposed, such as v grams [43], where instead of having a fixed length for all grams variable length grams were introduced, or the two-level q-gram inverted index [26]. In the literature “approximate string matching” also refers to the problem of finding a pattern string approximately in a text, which is the problem surveyed in Navarro’s paper [34] in 2001. An excellent tutorial on “approximate string search” topic [22]; q-grams based solution for the edit-distance metric has become the norm, see [3], [9], [11], [27], [28], [30], [32], [36], [43] and references therein.

The state-of-the-art in [30], [32] have conclusively shown that the q-grams based solution is the best method for the edit distance based metric. The q-grams based solution for edit-distance is also especially effective for the relatively short strings [30], which is the case for the problem (e.g. a large set of geo-tags rather than a long text document). That said, applying other metrics and/or other approximate string matching methods is definitely an interesting open problem to investigate. Another well-explored topic is the selectivity estimation of approximate string queries [10], [23], [24], [28], [29], [33]. Most of them use the edit distance metric and q-grams to estimate selectivity. Other work uses clustering [25]. Finally, special treatment was provided for selectivity of approximate string queries with small edit distance [28] and substring selectivity estimation was examined in [23], [29].

III. FEATURES OF THE MECHANISMS

A. Approximate string search

We refer to the problem of conjunctive keyword search with relaxed keywords as *approximate keyword search*. An important subproblem of approximate *keyword* search is that of approximate *string* search, defined as follows. Given a collection of strings, find those that are similar to a given query string. There are two main families of approaches to answer such queries. (1) Trie based method: The string collection is stored as a trie (or a suffix tree). For answer an approximate string query, we traverse the trie and prune subtrees that cannot be similar to the query. Popular pruning techniques use an NFA or a dynamic programming matrix with backtracking. (2) Inverted-index method: Its main idea is to decompose each string in the collection to small overlapping substrings (called grams) and build an inverted index on these grams. More details on fast indexing and search algorithms can be found in [12], [15], [16].

B. Spatial approximate-keyword search

Yao et al. [18] proposed a structure called MHR-tree to answer spatial approximate-keyword queries. They enhance an R-tree [6] with min-wise signatures at each node to compactly represent the union of the grams contained in objects of that subtree. They then use the concept of set resemblance between the signatures and the query strings to prune branches in the tree. The main advantage of this approach is that its index size does not require a lot of space since the min-wise signatures are very small. However, the method could miss query answers due to the probabilistic nature of the signatures.

Sattam Alsubaiee. et.al[2], we study how to support approximate keyword search on spatial data. Answering such queries efficiently is critical to these Web sites to achieve a high query throughput to serve many concurrent users. Although there are studies on both approximate keyword search and location-based search, the problem of supporting both types of search simultaneously has received little attention. To answer such queries, a natural index structure is to augment a tree-based spatial index with approximate-string indexes such as a gram-based inverted index or a trie-based index. The main focus of this work is a systematic study on how to efficiently combine these two types of indexes, and how to search the resulting index (called LBAK-tree) to find answers.

C. Nearest Neighbor and Top-k Queries

The processing of *k*-nearest neighbor queries (*k*NNs) in spatial databases is a classical subject. Most proposals use index structures to assist in the *k*NN processing. Perhaps the most influential *k*NN algorithm is due to Roussopoulos et al. [39]. In this solution, an R-tree[31] indexes the points, potential nearest neighbors are maintained in a priority queue, and the tree is traversed according to a number of heuristics. Other branch-and-bound methods modify the index structures to better suit the particular problem addressed [37], [41]. Hjaltason and Samet [35] propose an incremental nearest neighbor algorithm based on an R*-tree [22].

D. Location-Aware Text Retrieval Queries

Profitable search engines such as Google and Yahoo! have introduced local search services that appear to focus on the retrieval of local content, e.g., related to stores and restaurants. However, the algorithms used are not publicized. Much attention has been given to the problem of extracting geographic information from web pages. The extracted information can be used by search engines. McCurley [38] covers the notion of geo-coding and describes geographic indicators found in pages, such as zip codes and location names. Recent studies that consider location-aware text retrieval constitute the work most closely related to this study. Zhou et al. [47] tackle the problem of retrieving web documents relevant to a keyword query within a pre-specified spatial region. They propose three approaches based on a loose combination of an inverted file and an R*-tree. The best approach according to their experiments is to build an R*-tree for each distinct keyword on the web pages containing the keyword. As a result, queries with multiple keywords need to access multiple R*-trees and to intersect the results.

Building a separate R*-tree for each keyword also requires substantial storage.

Ian De Felipe.et.al, [17], the problem of top-k spatial keyword search is defined. The IR2-Tree is proposed as an efficient indexing structure to store spatial and textual information for a set of objects. Efficient algorithms are also presented to maintain the IR2-Tree, that is, insert and delete objects. An efficient incremental algorithm is presented to answer top-k spatial keyword queries using the IR2-Tree. We present a method to efficiently answer top-k spatial keyword queries, which is based on the tight integration of data structures and algorithms used in spatial database search and Information Retrieval (IR). In particular, this method consists of building an Information Retrieval R-Tree (IR2- Tree), which is a structure based on the R-Tree. At query time an incremental algorithm is employed that uses the IR2-Tree to efficiently produce the top results of the query.

IV. CONCLUSION

We reach the end of this survey on approximate string searching. Approximate string matching can be generally define as searching for substrings of a text that are within a predefined edit distance threshold from a given pattern. Our goal has been to present and explain the main ideas behind the existing techniques, to classify them according to the type of approach proposed, and to show how they perform in practice in a subset of possible practical scenarios.

REFERENCES

[1] Y.-Y. Chen, T. Suel, and A. Markowetz. Efficient query processing in geographic web search engines. In *SIGMOD*, 2006.

[2] S. Alsubaiee, A. Behm, and C. Li. Supporting location-based approximate-keyword queries. In *GIS*, pages 61–70, 2010.

[3] A. Arasu, S. Chaudhuri, K. Ganjam, and R. Kaushik. Incorporating string transformations in record matching. In *SIGMOD*, pages 1231– 1234, 2008.

[4] G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1), 2009.

[5] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, 2008.

[6] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD*, 1984.

[7] X. Cao, G. Cong, and C. S. Jensen. Retrieving top-k prestige-based relevant spatial web objects. *Proc. VLDB Endow.*, 3:373–384, 2010.

[8] K. Chakrabarti, S. Chaudhuri, V. Ganti, and D. Xin. An efficient filter for approximate membership checking. In *SIGMOD*, pages 805–818, 2008.

[9] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. Robust and efficient fuzzy match for online data cleaning. In *SIGMOD*, pages 313– 324, 2003.

[10] S. Chaudhuri, V. Ganti, and L. Gravano. Selectivity estimation for string predicates: Overcoming the underestimation problem. In *ICDE*, pages 227–238, 2004.

[11] S. Chaudhuri, V. Ganti, and R. Kaushik. A primitive operator for similarity joins in data cleaning. In *ICDE*, pages 5–16, 2006.

[12] M. Hadjieleftheriou, A. Chandel, N. Koudas, and D. Srivastava. Fast indexes and algorithms for set similarity selection queries. In *ICDE*, 2008.

[13] G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1):337–348, 2009.

[14] R. Hariharan, B. Hore, C. Li, and S. Mehrotra. Processing spatial-keyword (SK) queries in geographic information retrieval (GIR) systems. In *SSDBM*, 2007.

[15] C. Li, J. Lu, and Y. Lu. Efficient merging and filtering algorithms for approximate string searches. In *ICDE*, 2008.

[16] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1), 2001.

[17] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, pages 656–665, 2008.

[18] B. Yao, F. Li, M. Hadjieleftheriou, and K. Hou. Approximate string search in spatial databases. In *ICDE*, 2010.

[19] L. Gravano, P. G. Ipeirotis, H. V. Jagadish, N. Koudas, S. Muthukrishnan, and D. Srivastava. Approximate string joins in a database (almost) for free. In *VLDB*, pages 491–500, 2001.

[20] D. Zhang, B. C. Ooi, and A. K. H. Tung. Locating mapped resources in web 2.0. In *ICDE*, 2010.

[21] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma. Hybrid index structures for location-based web search. In *CIKM*, 2005.

[22] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *SIGMOD*, pp. 322–331, 1990.

[23] H. V. Jagadish, R. T. Ng, and D. Srivastava. Substring selectivity estimation. In *PODS*, pages 249–260, 1999.

[24] L. Jin and C. Li. Selectivity estimation for fuzzy string predicates in large data sets. In *VLDB*, pages 397–408, 2005.

[25] L. Jin, C. Li, and R. Vernica. Sepia: estimating selectivities of approximate string predicates in large databases. *The VLDB Journal*, 17(5):1213–1229, 2008.

[26] M.-S. Kim, K.-Y. Whang, J.-G. Lee, and M.-J. Lee. n-gram/2l: a space and time efficient two-level n-gram inverted index structure. In *VLDB*, pages 325–336, 2005.

[27] N. Koudas, A. Marathe, and D. Srivastava. Flexible string matching against large databases in practice. In *VLDB*, pages 1078–1086, 2004.

[28] H. Lee, R. T. Ng, and K. Shim. Extending q-grams to estimate selectivity of string matching with low edit distance. In *VLDB*, pages 195–206, 2007.

[29] H. Lee, R. T. Ng, and K. Shim. Approximate substring selectivity estimation. In *EDBT*, pages 827–838, 2009.

[30] C. Li, J. Lu, and Y. Lu. Efficient merging and filtering algorithms for approximate string searches. In *ICDE*, pages 257–266, 2008.

[31] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *SIGMOD*, pp. 47–57, 1984.

[32] G. Li, J. Feng, and C. Li. Supporting search-as-you-type using sql in databases. *TKDE*, To Appear, 2011.

[33] A. Mazeika, M. H. B¨ohlen, N. Koudas, and D. Srivastava. Estimating the selectivity of approximate string queries. *ACM TODS*, 32(2):12–52, 2007.

[34] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33:31–88, 2001.

[35] G. R. Hjaltason and H. Samet. Distance browsing in spatial databases. *ACM Trans. Database Syst.*, 24(2):265–318, 1999.

[36] S. Sahinalp, M. Tasan, J. Macker, and Z. Ozsoyoglu. Distance based indexing for string proximity search. In *ICDE*, pages 125–136, 2003.

[37] N. Katayama and S. Satoh. The SR-tree: an index structure for high-dimensional nearest neighbor queries. In *SIGMOD*, pp. 369–380, 1997.

[38] K. S. McCurley. Geospatial mapping and navigation of the web. In *WWW*, pp. 221–229, 2001.

[39] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *SIGMOD*, pp. 71–79, 1995.

[40] E. Sutinen and J. Tarhio. On using q-gram locations in approximate string matching. In *ESA*, pages 327–340, 1995.

[41] D. A. White and R. Jain. Similarity indexing with the SS-tree. In *ICDE*, pp. 516–523, 1996.

[42] E. Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.*, 92(1):191–211, 1992.

[43] X. Yang, B. Wang, and C. Li. Cost-based variable-length-gram selection for string collections to support approximate queries efficiently. In *SIGMOD*, pages 353–364, 2008.

[44] B. Yao, F. Li, M. Hadjieleftheriou, and K. Hou. Approximate string search in spatial databases. In *ICDE*, pages 545 – 556, 2010.

[45] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsuregawa. Keyword search in spatial databases: Towards searching by document. In *ICDE*, pages 688–699, 2009.

[46] D. Zhang, B. C. Ooi, and A. Tung. Locating mapped resources in web 2.0. In *ICDE*, pages 521–532, 2010.

[47] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma. Hybrid index structures for location-based web search. In *CIKM*, pp. 155–162, 2005.