

Efficient processing of continuous spatial-textual queries over geo-textual data stream

Kalpana Vivek Metre, Madan Kharat

Department of Computer Engineering, Faculty Information, MET's Institute of Engineering, Nashik, India

Article Info

Article history:

Received Jul 5, 2021

Revised Dec 9, 2021

Accepted Dec 22, 2021

Keywords:

Data stream

Edit-distance

Geo-textual

Locally optimal

Spatial-keyword queries

ABSTRACT

Due to the extensive use of social media and mobile devices, unbounded and massive data is generated continuously. The need to process this big data is increasing day by day. The traditional data processing algorithms fail to cater to the need of processing data generated by various applications such as digital geo-based advertising, and recommendation systems. There has been a high demand to process continuous spatial fuzzy textual queries over data stream of spatial-textual objects with high density by present location-based and social network-based service applications. For the spatial-keyword data stream, the performance plays a vital role as the geo information and keyword description matching is needed for every incoming streaming object. The various continuous geo-keyword query processing methods normally lack the support for fuzzy keyword matching when processing the objects from the geo-textual data stream. The edit distance-based approach with the adaptive partitioning tree index for the queries is used for fuzzy string matching and it outperforms than the existing approaches in storage cost and query performance cost.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Kalpana Vivek Metre

Department of Computer Engineering, MET's Institute of Engineering

Nashik-422003, India

Email: kvmetre@gmail.com

1. INTRODUCTION

Due to the increase in location-equipped devices and digital content generation, massive data is generated with textual and spatial information, described as spatial textual data. This data frequently come in streaming fashion in numerous applications viz social networks (e.g., Twitter and Facebook) and geo-enabled services (e.g., location-based digital advertising). It is communicated that millions of internet users are sending token-based data to Twitter. To comprehend spatial-textual data, it is essential to employ high-performance analytical methods. The main issue arises because of the processing of continuous spatial-textual queries instead of simple spatial-textual streams. It plays a pivotal role in many modern-day applications encompassing the dissemination of information [1], geo-based recommendation systems [2]. While processing geo-keyword queries over geo-textual continuous data, the rate of arrival of geo-textual data is rapid and queries are also dynamic, so the efficiency of execution of such types of queries plays a vital role. To address these issues, indexing for spatial-textual queries is needed which suits adaptability to both the keyword and spatial distributions of the submitted queries. As per the previous work in the related domain, two techniques are discussed [3], [4] to handle the processing of spatial-textual data with continuous queries on massive data. While processing geo-keyword queries, either the spatial-based filtering is considered followed by keyword filtering in query workload or keyword-based filtering first then followed by spatial filtering. In previous work, the R- tree structure is used for spatial representation of the queries, and

inverted lists are used for keyword storage. The solution for the same can be extended by using indexing for both keyword and spatial distributions in the workload of queries for higher efficiency. Chen *et al.* [3] the method of inverted indexing is used, but these are not efficient for textual filtering as the problem in consideration is a subset containment search in a textual outlook. Even though the inverted indexing techniques are established in spatial-keyword querying, they are mainly considered with superset containment queries [5], in which all query keywords contained within indexed objects are retrieved. Enhancement in performance is visible when the keywords' ordering is handled and combinations of multiple keywords are indexed. This can be seen in techniques defined for queries for superset containment using the ordered keyword trie [6].

Galić *et al.* [7] presented a framework that consists of the types of data and the operations supporting spatial-streaming data. A spatial-temporal language for queries is discussed [8] for processing geo-streaming data. A distributed framework under geo-streams is developed [9], [10] for efficiently supervising motile data objects using distributed geo-streaming data processing in massive clusters while in real-time. Although all these literary works are based only on the GeoStreams' spatial dimension, there is also a presence of textual information in geo streams. In force, large amounts of geo-spatial data generated recently also include geo-tagged micro-blogs, points of interest (POIs), images containing tags and geo-locations [3], [11]. As per the information in [12], approximately 30 million users submitted data with geo-tags on Twitter. Also, much such software like social networking sites (Facebook and Twitter) and regional services (regional advertising) send data in a rapid streaming pattern [12]. The hybrid indexing for the geo-spatial k-nearest neighbor (kNN) queries is discussed [13]. The authors used the tree approach and the space-filling curve concept for indexing the queries [14]. Various similarity measures to find the nearest objects are discussed [15]. The authors suggested a hybrid framework combining R-tree for location data and the inverted file for keyword purpose [16].

A new hybrid indexing method, adaptive geo-textual (spatial-textual) partition tree (AP-Tree), is coined to efficiently manage continuous and dynamic spatial-keyword queries. An AP-Tree can be defined as an f-ary tree in which there is a recursive division of queries based on keyword or spatial partitions (nodes). A cost model is also proposed to look over the assortment of division techniques so that the indexing is flexible with the query workload by integrating an alternative form of ordered keyword trie structure to improve the performance of textual filtering [12]. Nonetheless, the indexing for continuous spatial-keyword queries methods faces two fundamental problems in the present scenario. Firstly, these indexing approaches do not take into account partial keyword matching. In many applications, exact string matching does not cater to the need. Partial or fuzzy keyword matching is required for users who do not have a clear search condition, or some keyword errors (spelling errors), or when the queried data itself has some sort of unreliable fragments. This feature of approximate keyword matching is necessary for the processing of geospatial-based data, affirmed by the studies in [17]. The previous work for such type of query processing focused on mainly exact keyword matching.

Recently, the spatial-keyword search has been a sight of interest, which intends to fetch the related spatial-textual objects for a submitted spatial-keyword query. In general, the current work is a combination of spatial indexing and keyword indexing methods to construct objects so that incompatible objects are efficiently eliminated from textual and spatial perspectives. Broadly, these methods can be categorized into two classes: keyword-first [18]-[21] and spatial-first [16], [22]. In continuous query processing systems, there are quite a few endless queries that keep on running continuously. The incoming data objects are evaluated constantly and assigned to the matched queries logged in the system. Multiple works on publish/subscribe models explore a lot of continuous queries like predicate-based matching [23]-[26] and similarity-based ranking [27], [28]. However, spatial data is not taken into consideration. More recently, continuous dynamic spatial keyword queries are under study [29], [30], however, they look into continuous addressing of appropriate data objects with dynamic queries, making it fundamentally incompatible with the issue. However, it is proposed a time-based publish/subscribe system while taking into account both textual and spatial factors and semantic-based top-k search is performed [31].

Here, it explores the most notable works encompassing indexing techniques for geo-textual data. The techniques of indexing of geo-textual data are broadly categorized into two approaches, i.e., for static data and dynamic streaming data. In the spatial data approach, all of the spatial data is being retrieved from a spatial database with each object defined by a set of keywords. The location information and keyword information are to be stored in all incoming objects. These techniques store both the spatial information and keywords of every data object to execute spatial-keyword queries. The data structure R-tree has been greatly used to ground geo-textual data in these methods. For example, in the work [32], the authors looked into a hybrid and combined indexing structure that manages inverted lists for the document tokens and also incorporated an extended R-trees data structure for geo-textual tokens. Along the same lines, Zhang *et al.* [33] put forward an bR*-tree, an extension of R*-tree with bitmaps. Also, various spatial indexing techniques have been explored for geo-keyword data. An inverted linear Quadtree (IL-Quad tree) [20] formed on an inverted index. The linear

Quadtree was proposed for top-k search based on location and keyword. The k-dimensional (KD) tree structure with an inverted list was proposed for the indexing of geo-textual data [34].

The methods used in the indexing of static spatial-textual data will not suffice to incorporate the dynamic geo-textual data. Hence, in recent years, different query indexing attempts, which work on the indexing of continuous queries for the filtering process of geo-textual data are being looked into to tackle this issue. These methods of indexing are roughly classified into three categories: indexing on keyword-first, indexing on spatial-first and adaptive indexing. The inverted file quad-tree (IQ-tree) [3] uses a Quadtree to classify queries such that every query is connected with one or more Quadtree grids/cells. All the queries in every cell are allocated to the posting list having its frequent keyword using a ranked-key-based inverted list. In the same way, the R^l-tree i.e., R-tree with tokens [4] uses the R-tree first for indexing the queries considering their spatial data, and then every R-tree node stores the keywords of its child queries for appropriately filtering the textual data. The spatial data feature is the only determining factor during the construction of the tree structures in IQ-Tree [3] and R^l-tree [4]. Thus, they take a hit in the overall performance as they use separate keyword and spatial distributions in the submitted queries. To tackle this, the authors proposed an adaptive textual-spatial partition tree data structure i.e., AP-tree [12] utilizes an f-ary data structure that processes the indexing of the queries in a flexible way considering the workload of queries. But the problem lies in the fact that the current indexing techniques for continuous and dynamic spatial-keyword queries do not support approximate or partial keyword searches. The authors put forth an MHR Tree i.e., R Tree using a min-wise signature with linear hashing which combines the R-tree and Min-wise signature to execute spatial keyword queries where the objects are searched for approximate string matching [17]. The authors used the advanced adaptive partitioning tree concept in which q-grams' signatures are stored in an advanced adaptive partition tree-AAP tree (AP-Tree+).

The dynamic programming approach is used to dividing keywords into different partitions [35]. The sub-queries are generated from continuous queries using a simulated annealing algorithm. The push and pull-based data dissemination approaches are used to send changes to the user as per the user's interest [36]. A significant amount of temporal data is generated in network monitoring, and stock exchange. Which can be used in online decision making as dynamic data items are generated [37]. A presorted-nearest index tree algorithm is introduced for nearest neighbor queries on mobile objects within desired period and outperformed in saving time compared to KD-tree approach [38]. Vantage point tree indexing with spectral clustering is used for high dimensional data for effective data retrieval for user query [39]. The authors discussed various real time classification and clustering techniques for data stream as well as platforms for mining of data streams [40]. A new framework for location-based services under the umbrella of mobile cloud computing providing the location privacy and integrity of service results is discussed [41].

As the incoming query data is massive in number, the need to come up with effective indexing techniques so that quite a few incompatible queries can be refined at a nominal cost has become vital. In this paper, for the processing of spatial keyword temporal region queries, a Levenshtein distance adaptive partition tree (LDAP tree) is proposed. In the LDAP tree, the keywords are stored for approximate string matching. A locally optimal method is used for keyword partitioning and the Levenshtein edit distance concept is used for approximate string matching. The paper emphasizes the challenges of approximate keyword searching and providing an efficient solution for streaming data.

2. PROPOSED METHOD

For the processing of spatial keyword temporal region queries, a LDAP tree is proposed containing keyword nodes, spatial nodes and query nodes. A locally optimal greedy method is used for keyword partitioning on keyword node. In LDAP tree, the keywords are stored for approximate string matching.

2.1. Problem formulation

The basics of spatial keyword temporal data stream and spatial keyword temporal queries are provided.

- Definition 1: Spatial keyword temporal object is defined as $O = (id, t, loc, kws)$ where t is a timestamp and loc is a location of the object. The set of keywords associated with the object is denoted by kws .
- Definition 2: Spatial keyword temporal data stream is defined as $S = \{O_i \mid i \in [1, +\infty] \wedge O_i.t \leq O_{i+1}.t\}$. It is an unbounded set of spatial-keyword objects in increasing timestamp order.
- Definition 3: The continuous spatial-keyword temporal query is denoted as $q = (id, t_1, t_2, r, kw)$ where t_1 and t_2 are the timestamps during which the query exists and r is the rectangular region of the query. The kw is a set of distinct keywords associated with the query. A Spatial-keyword temporal object O in stream S matches the query q if the three conditions, as stated in (1), are fully satisfied.

$$o.loc \in q.r \text{ and } q.t_1 \leq o.t \leq q.t_2 \text{ and } q.kw \subseteq o.kws \text{ and } sim(q.kw, o.kws) \geq sthr \quad (1)$$

In consequent sections, continuous spatial-keyword-temporal query and spatial-textual-temporal object are abbreviated as query and object respectively. It is assumed that the keywords in the queries and objects are in sorted order such that $W_i < W_j$ if $i < j$ where W_i and W_j are keywords in queries and objects.

For example: There are four spatial-keyword queries $\{q_1, q_2, q_3, q_4\}$ and two objects $\{O_1, O_2\}$. The keyword set for O_1 is $\{W_1, W_3, W_6\}$ and O_2 is $\{W_1, W_3, W_5\}$. So, the object O_1 satisfy the spatial and keyword condition for q_1 as O_1 contain all the keywords of q_1 . The object O_2 satisfy the spatial and keyword condition for q_4 as shown in Figure 1.

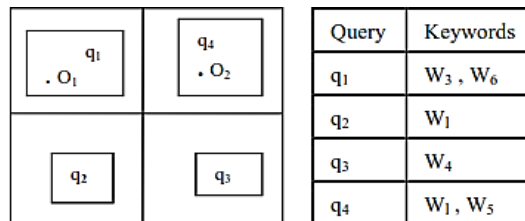


Figure 1. The example for set of spatial-keyword queries

3. METHOD

3.1. Continuous query processing using LDAP tree

The architecture of the system is as shown in Figure 2 is: i) Continuous Query Submission: The continuous queries are given by the user as per the definition stated in section 2.1. ii) Construction of Index for Query/Update: The hybrid indexing approach LDAP Tree is used for indexing the queries. For dividing the queries into different subgroups, the local optimal selection based greedy approach is used. The construction and updating of the index will be performed. iii) Acquisition of data stream containing objects as stated in section 2.1. vi) The objects will be mapped to the index for searching relevant queries using 3 predicates as stated in section 2.1. It will check for locality constraint, temporal constraint, and matching of the keyword for approximations rather than exact. The edit distance (Levenshtein distance) based approach is used for fuzzy matching of keywords. v) The relevant queries for the object will be returned.

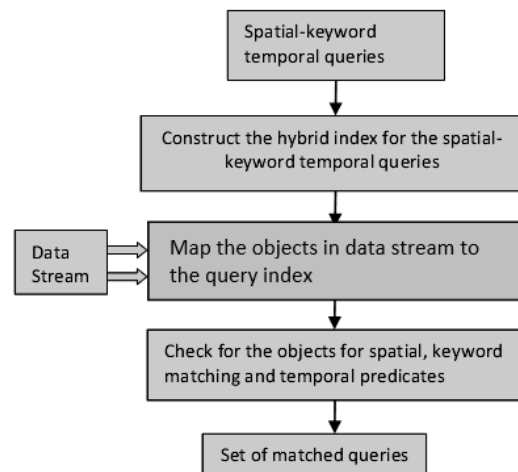


Figure 2. Architecture for evaluation of spatial-textual region queries

3.2. Spatial and keyword filtering framework

For indexing spatial-keyword-temporal queries and searching the object from the unbounded data stream, Levenshtein distance-based adaptive spatial-keyword filtering approach (LDAP) is used as shown in Figure 3. The two-way partitioning is used for spatial filtering and keyword filtering. The queries are recursively partitioned in a top-down manner using spatial division and keyword division. LDAP tree comprises of three types of nodes: spatial node, keyword node, and query node. A node in a tree can be a keyword node or spatial node depending on the method of partitioning used. The term cut c is used to define

the degree of the intermediate node. The queries will be stored at the leaf node of the tree. Each query can be present in one or more query leaf nodes as per its set of keywords and the geo region. The construction of an index for spatial-keyword-temporal queries can be explained in detail by considering spatial partitioning and keyword partitioning in detail [12].

- Spatial node and its partitioning: The spatial region are recursively subdivided into smaller c spatial nodes. The region of the spatial node N is denoted by N_r which is subdivided again into c spatial grid nodes. A query q on that spatial node N is assigned to cut or cell C if query region $q.r$ contains C or overlaps C . The query can be allocated to multiple spatial child nodes.
- Keyword node and its partitioning: Assuming the ordering of keywords in the vocabulary (set of all distinct keywords), in each query. For the generation of the cuts c for the given keyword node, the queries allocated to that node are divided into c ordered cuts according to the index of that keyword in the query and that index will decide the partition offset of that keyword node N . Each cut or bucket B may contain the set of ordered keywords $C[K_i, K_j]$ where K_i and K_j are keywords for left and right boundaries in the given cut. For dividing the keyword node into different cuts, a cost model is used. The cost of partition P is computed as [12]:

$$C(P) = \sum_{i=1}^C w(B_i) * p(B_i) \tag{2}$$

where C : number of cuts and $w(B_i)$: number of queries associated to B_i .

$$p(B) = \sum_{w \in B} p(w) \tag{3}$$

$$p(w) = \frac{\text{freq}(w)}{\sum_{w \in P} \text{freq}(w)} \tag{4}$$

$$p(B) = \text{area}(B) / \text{area}(N) \tag{5}$$

- If q contains keywords $\{W_1, W_2, \dots, W_n\}$, the first m keywords (where $1 \leq m \leq n$) in $\{W_1, W_2, \dots, W_n\}$ will be explored in previous keyword nodes only if there are m number of keyword nodes in the q -node's ancestor nodes including parent node. So, only last $(n-m)$ keywords for each query will be stored. Where KW and SP denote keyword node and spatial node respectively.

As discussed in Algorithm 1, the greedy approach is used for the keyword partition of query index where all the keywords are divided into c partitions/cuts. Algorithm 2 describes the procedure of building an index for query dataset Q where where keyword and spatial nodes is generated using cost model. The dummy cut/cell is used for the queries if the number of keywords in a query is less than the offset of the partition or the region of the query contains the region of the spatial node. The object searching is done using Algorithm 3 on query index where the searching is done from the root node and every keyword in the incoming object is searched in each partition of keyword node. In case of spatial node, the location of the object is checked in the cells of spatial node.

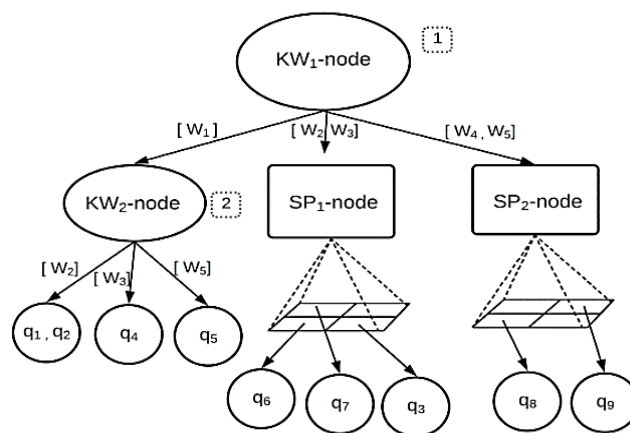


Figure 3. LDAP tree for query indexing

Algorithm 1: GreedyMethod_for_keyword Partition (KW, c)

Input: KW: Set of keywords, c: fanout of node i.e., cuts

Output: Pc: Partition with keywords with c cuts

```

1) find an initial solution ← partition Pc which partitions KW into c cuts with similar weights;
2) for (2<=j<=c) do
3)   for every keyword u lies between left (Cj-1) and right (cj) do
4)     calculate C(Pc) // using cost model
5)     if (lower C(Pc) is achieved)
6)       Update cuts Cj-1 and cj in Pc
7) return Pc

```

Algorithm 2: Index_construct (CN, Q, l, Pk, Ps)

Input Q: Query dataset, CN: current node, l: offset for partitioning keywords

Ps and Pk: flags for spatial and keyword partitions respectively, thr: threshold value for storing queries in query node

Output: LDAP-Tree

```

1 if (Pk is not true and Ps is not true) or |Q| < thr, then
2   CN is a query-node for set of queries Q
3   return
4 k-cost = +∞ s-cost = +∞
5 if (Pk is true) then
6   k-cost ← compute cost for partition on keywords on Q having offset l
7 if (Ps is true) then
8   s-cost ← compute cost for partition on spatial on Q
9 if (Pk is selected (i.e., k-cost < s-cost) then
10  CN is a keyword-node having offset N1 = l
11  create dummy cut for Queries Qd where |q.kw| < l
12  for every child node B (i.e., cut) of node CN do
13    QB = Q - Qd
14    Index_construct(B, QB, l + 1, Pk, Ps);
16 if CN is a spatial-node
17   Qd = queries from Q contains N2
18   create dummy cut for Queries Qd
19   for every child node B (cell) of node CN do
20     QB = Q - Qd
21     index_construct(B, QB, l, Pk, Ps);

```

Algorithm 3: ObjectSearch (o, s, CN)

Input: o: incoming spatial-keyword object, s: start position wrt o.kws, CN: current accessed node

Output: A: Result set containing the matched and relevant queries for an object

```

1 if CN is a query-node then
2   check_queries(N)
3   update A
3 return
4 if CN is a keyword-node then
5   for s <= j <= o.kws do
6     find appropriate cut based on Kj in (o.kws)
7     if any cut is not explored then
8       ObjectSearch(o, j + 1, cut)
9 if exist (dummy_cell) then
10  ObjectSearch(o, s, dummy_cell)
11 else
12  find the grid that covers o.loc using cell structure;
13  ObjectSearch(o, s, cell)
14  if dummy_cut is present then
15  ObjectSearch(o, s, dummy_cut)

```

In the algorithm check_queries(), the following conditions are checked.

- 1) The superset containment of keywords in queries to keywords in the object.
- 2) The object location lies in the region of query and time constraint is checked.
- 3) The approximate matching of the keyword in query and the keyword object is done using Levenshtein distance method at query node as well as keyword node for every query keyword.

4. RESULTS AND DISCUSSION

The results of experimental evaluation are presented. Experimental Setup: The geo-keyword-based services have been frequently used in numerous applications e.g., social media, and digital advertising. To evaluate the discussed indexing approach, the following datasets are used.

- i) AIS dataset contains the geo-locations which are taken from Chorochronos Archive (<http://www.chorochronos.org>) [35].

- ii) Keywords are taken from Newsgroups with about 61,000 keywords (<http://people.csail.mit.edu/ljrennie/120Newsgroups>) [35]. Object dataset is generated using the spatial locations and keywords from vocabulary (Newsgroups): 500,000 with 1-9 keywords.
- iii) To generate query workload: 100,000 spatial-keyword objects from the dataset are selected. From every selected object, the keywords for queries are selected randomly from the selected objects varying it between 1 and 5. The query region of the query is taken as rectangle with center as the spatial location of the object. The region size is selected as 0.001. The experiments are run on i7-7500U @2.70 GHz with 8 GB RAM.
- iv) Parameters of Experiments: Query dataset is of size 10000, thr (the maximum no. of queries in a query node) =40, no. of partitions $c=200$, $sth=0.5$.

The results i.e., query execution time and storage cost for query index is compared with the methods used in [35] as shown in Figure 4 and Figure 5. To evaluate the performance of the queries on data stream, which is continuous, where the queries are also dynamic, the two measures are considered i.e., query execution cost and storage cost for query indexing. As the keywords are stored instead of q-gram signatures in the indexing data structure for the queries, it reduces the storage cost of the index significantly. During the object searching, the fuzzy keyword matching is done based on the edit distance (Levenshtein distance) method as the keywords are short text descriptions. It results in faster execution time.

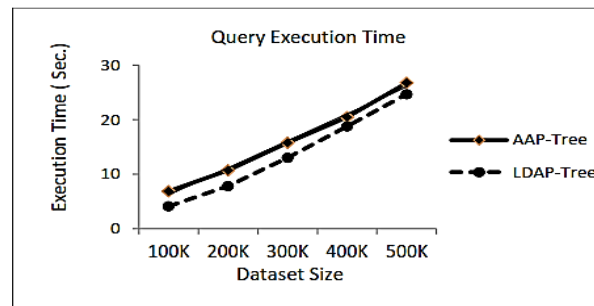


Figure 4. Comparison of query execution time

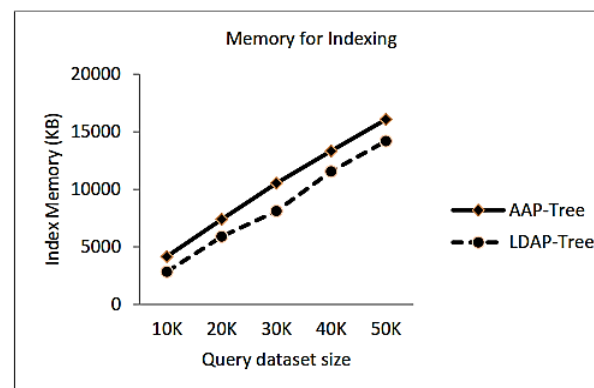


Figure 5. Storage cost for indexing query dataset

5. CONCLUSION

It is necessary to cater to the increasing need for processing spatial-keyword-temporal data efficiently which is generated continuously due to the extensive use of social media and web search. The index for finite queries was generated for efficient execution of spatial-keyword queries which are less dynamic than the streaming data. The approximate string matching suited to these applications rather than exact string matching. The edit distance-based approach was used for fuzzy string matching which is suitable for short text descriptions with typo errors. The LDAP tree outperformed in storage costs for queries and execution cost for object searching than the previous methods. The approach can be extended to multiple streams and distributed query processing.




REFERENCES

- [1] T. W. Yan and H. García-Molina, "Index structures for selective dissemination of information under the Boolean model," *ACM Transactions on Database Systems*, vol. 19, no. 2, pp. 332–364, Jun. 1994, doi: 10.1145/176567.176573.
- [2] M.-H. Park, J.-H. Hong, and S.-B. Cho, "Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices," in *Ubiquitous Intelligence and Computing*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1130–1139.
- [3] L. Chen, G. Cong, and X. Cao, "An efficient query indexing mechanism for filtering geo-textual data," in *Proceedings of the 2013 international conference on Management of data - SIGMOD '13*, 2013, p. 749, doi: 10.1145/2463676.2465328.
- [4] G. Li, Y. Wang, T. Wang, and J. Feng, "Location-aware publish/subscribe," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, Aug. 2013, pp. 802–810, doi: 10.1145/2487575.2487617.
- [5] S. Helmer and G. Moerkotte, "A performance study of four index structures for set-valued attributes of low cardinality," *The VLDB Journal The International Journal on Very Large Data Bases*, vol. 12, no. 3, pp. 244–261, Oct. 2003, doi: 10.1007/s00778-003-0106-0.
- [6] Z. Hmedeh, H. Kourdounakis, V. Christophides, C. du Mouza, M. Scholl, and N. Travers, "Subscription indexes for web syndication systems," in *Proceedings of the 15th International Conference on Extending Database Technology - EDBT '12*, 2012, p. 312, doi: 10.1145/2247596.2247634.
- [7] Z. Galić, M. Baranović, K. Krizanović, and E. Mešković, "Geospatial data streams: Formal framework and implementation," *Data & Knowledge Engineering*, vol. 91, pp. 1–16, May 2014, doi: 10.1016/j.datak.2014.02.002.
- [8] S. Eom, S. Shin, and K.-H. Lee, "Spatiotemporal query processing for semantic data stream," in *Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing (IEEE ICSC 2015)*, Feb. 2015, pp. 290–297, doi: 10.1109/ICOSC.2015.7050822.
- [9] Z. Galić, "Spatio-Temporal Data Streams and Big Data Paradigm," 2016, pp. 47–69.
- [10] Z. Galić, E. Mešković, and D. Osmanović, "Distributed processing of big mobility data as spatio-temporal data streams," *Geoinformatica*, vol. 21, no. 2, pp. 263–291, Apr. 2017, doi: 10.1007/s10707-016-0264-z.
- [11] X. Chen *et al.*, "Design Automation for Interwell Connectivity Estimation in Petroleum Cyber-Physical Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 2, pp. 255–264, Feb. 2017, doi: 10.1109/TCAD.2016.2584065.
- [12] X. Wang, Y. Zhang, W. Zhang, X. Lin, and W. Wang, "AP-Tree: Efficiently support continuous spatial-keyword queries over stream," in *2015 IEEE 31st International Conference on Data Engineering*, Apr. 2015, pp. 1107–1118, doi: 10.1109/ICDE.2015.7113360.
- [13] K. V. Metre and M. U. Kharat, "Scalable Execution of KNN Queries using Data Parallelism Approach," *International Journal of Engineering & Technology*, vol. 7, no. 4.19, p. 1060, Nov. 2018, doi: 10.14419/ijet.v7i4.19.28286.
- [14] K. V. Metre, "Location based Continuous Query Processing over Geo-streaming Data," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 1S, pp. 106–114, Apr. 2021, doi: 10.17762/turcomat.v12i1S.1583.
- [15] M. U. Kharat, R. Dahake, and K. V. Metre, "Clustering techniques for content-based feature extraction from image," 2018, pp. 100–121.
- [16] G. Cong, C. S. Jensen, and D. Wu, "Efficient retrieval of the top-k most relevant spatial web objects," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 337–348, Aug. 2009, doi: 10.14778/1687627.1687666.
- [17] F. Li, B. Yao, M. Tang, and M. Hadjieleftheriou, "Spatial Approximate String Search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1394–1409, Jun. 2013, doi: 10.1109/TKDE.2012.48.
- [18] M. Christoforaki, J. He, C. Dimopoulos, A. Markowetz, and T. Suel, "Text vs. space," in *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, 2011, p. 423, doi: 10.1145/2063576.2063641.
- [19] J. B. Rocha-Junior, O. Gkorgkas, S. Jonassen, and K. Nørvag, *Efficient processing of top-k spatial keyword queries*, Internatio. Springer, 2011.
- [20] Chengyuan Zhang, Ying Zhang, Wenjie Zhang, and Xuemin Lin, "Inverted linear quadtree: Efficient top k spatial keyword search," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, Apr. 2013, vol. 28, no. 7, pp. 901–912, doi: 10.1109/ICDE.2013.6544884.
- [21] C. Zhang *et al.*, "Diversified spatial keyword search on road networks," in *Advances in Database Technology-EDBT 2014: 17th International Conference on Extending Database Technology, Proceedings*, 2014, pp. 367–378, doi: 10.5441/002%2Fedbt.2014.34.
- [22] I. De Felipe, V. Hristidis, and N. Rische, "Keyword Search on Spatial Databases," in *2008 IEEE 24th International Conference on Data Engineering*, Apr. 2008, pp. 656–665, doi: 10.1109/ICDE.2008.4497474.
- [23] F. Fabret, H. A. Jacobsen, F. Llirbat, J. Pereira, K. A. Ross, and D. Shasha, "Filtering algorithms and implementation for very fast publish/subscribe systems," *ACM SIGMOD Record*, vol. 30, no. 2, pp. 115–126, Jun. 2001, doi: 10.1145/376284.375677.
- [24] S. E. Whang *et al.*, "Indexing Boolean expressions," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 37–48, Aug. 2009, doi: 10.14778/1687627.1687633.
- [25] M. Sadoghi and H.-A. Jacobsen, "BE-tree," in *Proceedings of the 2011 international conference on Management of data - SIGMOD '11*, 2011, p. 637, doi: 10.1145/1989323.1989390.
- [26] D. Zhang, C.-Y. Chan, and K.-L. Tan, "An efficient publish/subscribe index for e-commerce databases," *Proceedings of the VLDB Endowment*, vol. 7, no. 8, pp. 613–624, Apr. 2014, doi: 10.14778/2732296.2732298.
- [27] K. Mouratidis and H. Pang, "Efficient Evaluation of Continuous Text Search Queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 10, pp. 1469–1482, Oct. 2011, doi: 10.1109/TKDE.2011.125.
- [28] A. Shraer, M. Gurevich, M. Fontoura, and V. Josifovski, "Top-k publish-subscribe for social annotation of news," *Proceedings of the VLDB Endowment*, vol. 6, no. 6, pp. 385–396, Apr. 2013, doi: 10.14778/2536336.2536340.
- [29] D. Wu, M. L. Yiu, C. S. Jensen, and G. Cong, "Efficient continuously moving top-k spatial keyword query processing," in *2011 IEEE 27th International Conference on Data Engineering*, Apr. 2011, pp. 541–552, doi: 10.1109/ICDE.2011.5767861.
- [30] W. Huang, G. Li, K.-L. Tan, and J. Feng, "Efficient safe-region construction for moving top-K spatial keyword queries," in *Proceedings of the 21st ACM international conference on Information and knowledge management - CIKM '12*, 2012, p. 932, doi: 10.1145/2396761.2396879.
- [31] L. Chen, G. Cong, X. Cao, and K.-L. Tan, "Temporal Spatial-Keyword Top-k publish/subscribe," in *2015 IEEE 31st International Conference on Data Engineering*, Apr. 2015, pp. 255–266, doi: 10.1109/ICDE.2015.7113289.
- [32] R. Göbel, A. Henrich, R. Niemann, and D. Blank, "A hybrid index structure for geo-textual searches," in *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM '09*, 2009, p. 1625, doi: 10.1145/1645953.1646188.
- [33] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsuregawa, "Keyword Search in Spatial Databases: Towards Searching by Document," in *2009 IEEE 25th International Conference on Data Engineering*, Mar. 2009, pp. 688–699, doi: 10.1109/ICDE.2009.77.




- [34] J. Li, H. Wang, J. Li, and H. Gao, "Skyline for geo-textual data," *GeoInformatica*, vol. 20, no. 3, pp. 453-469, Jul. 2016, doi: 10.1007/s10707-015-0243-9.
- [35] Z. Deng *et al.*, "An Efficient Indexing Approach for Continuous Spatial Approximate Keyword Queries over Geo-Textual Streaming Data," *ISPRS International Journal of Geo-Information*, vol. 8, no. 2, p. 57, Jan. 2019, doi: 10.3390/ijgi8020057.
- [36] M. Thombare and K. V. Metre, "Query Optimization and Execution of Dynamic Data Items in Network Aggregation Environment," *Elsevier*, pp. 1406-1413, 2014.
- [37] M. Thombare and K. V. Metre, "Aggregation Environment for Query Optimization in Network Monitoring," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 2, no. 6, pp. 1515-1518, 2014.
- [38] T. T. Zan and S. Phyu, "Mobile Location Indexing Based On Synthetic Moving Objects," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, p. 2556, Aug. 2019, doi: 10.11591/ijece.v9i4.pp2556-2563.
- [39] P. R. and K. M. Sundaram, "Spectral Clustering and Vantage Point Indexing for Efficient Data Retrieval," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 4, p. 2261, Aug. 2018, doi: 10.11591/ijece.v8i4.pp2261-2271.
- [40] E. Alothali, H. Alashwal, and S. Harous, "Data stream mining techniques: a review," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 2, p. 728, Apr. 2019, doi: 10.12928/telkonnika.v17i2.11752.
- [41] Y. Yan, H. Xiaohong, and W. Wanjun, "Location-Based Services and Privacy Protection under Mobile Cloud Computing," *Bulletin of Electrical Engineering and Informatics*, vol. 4, no. 4, pp. 345-354, Dec. 2015, doi: 10.11591/eei.v4i4.548.

BIOGRAPHIES OF AUTHORS



Kalpana Vivek Metre    is working in MET's Institute of Engineering, Nashik, India. She has teaching experience of 20+ years. She has published/presented 40+ in national and international journals and conferences resp. She has guided UG as well as PG students for project work. Her area of interest is algorithms, data science, and database. She can be contacted at email: kvmetre@gmail.com.



Madan Kharat    is working as Head of Department and Professor in Department of Computer Engineering, MET's Institute of Engineering, Nashik, India. He has teaching experience of 27+ years. He has published/presented numerous research papers in national and international journals and conferences resp. He has guided Ph. D. scholars, PG students, and UG students. His area of interest is networking, image processing, wireless networks, and signal processing. He can be contacted at email: mukharat@rediffmail.com.