Journal of Ambient Intelligence and Smart Environments 12 (2020) 281–312
DOI 10.3233/AIS-200565
IOS Press

# Indexing of real time geospatial data by IoT enabled devices: Opportunities, challenges and design considerations

Natalia Chaudhry [a,*], Muhammad Murtaza Yousaf [a] and Muhammad Taimoor Khan [b]
[a] *PUCIT, University of the Punjab, Lahore, Pakistan*
*E-mails: natalia@pucit.edu.pk, murtaza@pucit.edu.pk*
[b] *School of Computing and Mathematical Sciences, University of Greenwich, London, UK*
*E-mail: m.khan@gre.ac.uk*

**Abstract.** We are moving towards 'smart' world in which industries, such as healthcare, smart cities, transportation, and agriculture have started using IoT (Internet of Things). These applications involve huge number of sensors and devices that generate high volume of real time data. To perform useful analytics on this data, location and spatial awareness characteristics of devices need to be considered. Wide range of location-based services and sensors in GIS have to manage moving objects that change their position with respect to time. These applications generate voluminous amount of real time geospatial data that demands an effective query processing mechanism to minimize the response time of a query. Indexing is one of the traditional ways to minimize the response time of a query by pruning the search space. In this paper, we performed a detailed survey of the literature regarding the indexing of real time geospatial data generated by IoT enabled devices. Some major challenges relevant to indexing of moving objects are highlighted. Various important index design considerations are also discussed. The goal is to help researchers in understanding the principles, methods, and challenges in the indexing of real time geospatial data. This will also aid in identifying the future research opportunities.

Keywords: Spatial indexing, concurrency control, GIS, Internet of things, spatiotemporal data

## 1. Introduction

IoT characterize the future in which devices are connected to each other using the Internet and allow human-to-machine and machine-to-machine interaction. IoT involves large number of devices connected with each other to capture enormous amount of data. Processing this huge amount of data is a crucial aspect in performing a valuable spatiotemporal analytics. Nowadays, idea of Bring Your Own Device (BYOD) has been propagated that has revolutionized the way devices are used. With extensive use of location-based services and sensors, geography has been inevitably linked with modern technologies. Smart building devices, car navigation systems, and autonomous vehicles must be equipped with location intelligence to function properly. For example, in order for smartphones to correctly predict the directions they must have to embed spatial awareness within them.

Wide range of IoT based applications have to manage spatial objects that change their position with respect to time. Examples of such objects include oil tankers, cars, and air planes. These objects are used in number of GPS and location based applications including crowd tracking, robot path planning, and traffic monitoring. Utilization of such spatiotemporal data plays an important role in addressing many societal issues. Majority of applications in the domain of transportation, military, agriculture, and business acquire massive amount of spatiotemporal data from sensors

[*] Corresponding author. E-mail: natalia@pucit.edu.pk.

that need effective management. Indexing techniques emerged as a solution to prune the search space and minimize the response time of spatial query operations.

This paper discusses the challenges faced during spatial query processing. We begin with describing an integration of GIS and IoT, the query model involved in this integration, and brief background of spatial data processing. Challenges, design considerations, and research opportunities are discussed in the context of spatial query processing. In many real life applications such as Didi, the number of queries approach up to several millions during peak hours. This makes spatial query processing a challenging task. The sooner the queries are entertained the better will be the validity of results returned. For example, a quick response time is needed in a disaster management applications such that appropriate measures are taken on time. Many spatial indexing methods have been designed to date. But, all of them have not addressed some critical challenges in spatial index design and query processing. Particularly, in case of indexing of moving objects, the read/write query conflicts have not been addressed. To design an indexing solution, one must focus on some important design considerations and challenges to make the solution suitable for real life applications. Therefore, it is significant to have a study in place that discusses these design considerations and challenges.

### 1.1. Difference from existing surveys

To the best of our knowledge, there is no study in place that has thoroughly reviewed the domain of indexing of real time geospatial data generated by IoT devices and highlighted the challenges, index design considerations, and research opportunities. This paper has covered all of these shortcomings of existing research works. This survey is focused on reviewing the literature and domain of spatial indexing.

The survey [100] provided a summarized view of recent research efforts made on spatial data mining. Background of spatial data mining is presented while briefly discussing the structure of spatial data and data structures involved in handling it. The discussion is focused on machine learning based methods for knowledge discovery in spatial databases. Generalization, clustering, and many other variants are discussed in the light of spatial data mining. The challenges in spatial query processing has not been put into focus.

The paper [45] surveyed the whole big spatial data domain briefly. As the focus of this survey is not on some specific module of big spatial data domain (e.g., spatial query processing), therefore various aspects of big spatial data processing, namely, visualization, underlying architecture, implementation approaches, and indexing are reviewed very briefly. Only some recent works have been highlighted with regard to each component.

In [221], a brief overview of modeling and management of spatiotemporal data is presented. The discussion includes only some of the research efforts made for modeling and management of moving objects. However, the discussion of challenges in query processing with moving objects and description of landscape of literature is missing.

The spatiotemporal access methods used for indexing past, current, and future locations of moving objects have been discussed in [125]. However, the challenges with each data access methods have not been discussed.

In view of the above discussion, the need for a survey on spatial indexing is obvious. Most of the existing surveys are focused on the review of big spatial data domain. The focus of survey [100] is on spatial data mining. Spatial indexing and query processing has not been discussed in this survey. The research work [45] reviewed some spatial indices very briefly but didn't consider spatiotemporal data in its discussion. The research works, [221] and [125] have not extensively evaluated spatial indices. As opposed to these research works, we have provided a complete overview to the domain of spatiotemporal indexing. We discussed the challenges in query processing in this domain. We have also extensively reviewed the literature starting from early 90s till to date. Our main contributions are presented as follows.

- Detailed review of background on spatiotemporal query processing.
- The significance of spatial indexing is highlighted and extensive literature review is done.
- Comparative review of the literature is performed on spatial indexing with regard to number of view points and design considerations.
- Challenges and research opportunities related to spatiotemporal query processing and indexing are discussed.

The rest of the document is organized as follows: Section 2 introduces the domain of IoT and spatiotemporal data processing and various major concepts covered by these domains. Section 3 reviews the existing research works related to the spatial querying and in-

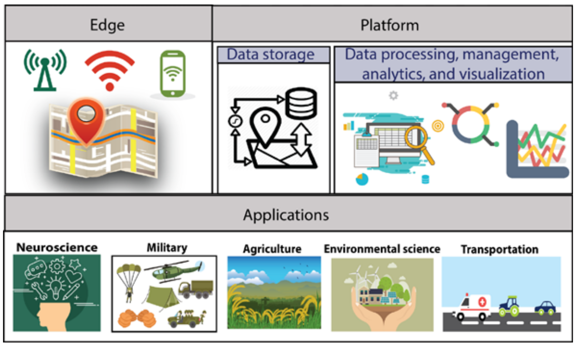

Fig. 1. Layered architecture of IoT based solution.

dexing of moving objects. Section 4 describes the challenges associated with spatial indexing. Section 5 describes the important design considerations relevant to spatial indexing. Section 6 discusses research opportunities. Section 7 concludes the paper.

## 2. Integration of IoT and GIS

IoT data is unstructured and has heterogeneous formats as it is usually acquired from diverse sources. Huge amount of data with spatiotemporal characteristics will be generated by these diverse sensors. This has made information representation, retrieval, management, and processing of spatiotemporal data a complex process. Typical IoT based solution has three layers: 1) *edge technology* to acquire data using embedded technology, 2) *platform* involved in manipulation of data and extraction of intelligence from it, and 3) *applications* that use this intelligence. Figure 1 depicts this layered architecture. At edge, there are sensors, actuators, and devices for sensing the environment. Platform layer involves processing, management, and analytics aspects.

The scope of this paper lies within layer 2 in which processing of spatiotemporal data with moving objects will be considered. Typically, the query model involves sensors or GPS based devices deployed in the environment and moving objects that report their location to the server continuously. This has been depicted in Fig. 2.

Processing of this spatiotemporal data should consider the crucial aspects and issues in IoT domain, such as limited battery life of sensors. With limited battery life, sensors would not be able to do extensive processing. Therefore, it is important to minimize the query processing time and communication among sensors and the server. The primary focus of this paper is therefore on query processing and indexing of spatiotemporal data generated by sensors distributed in the environment. Indexing is essential to minimize the processing costs in platform layer of IoT. The upcoming sections introduce the domain of spatial query processing and indexing and major concepts and applications involved in it.

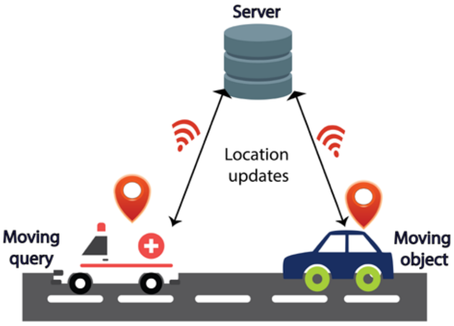

Fig. 2. Query model.

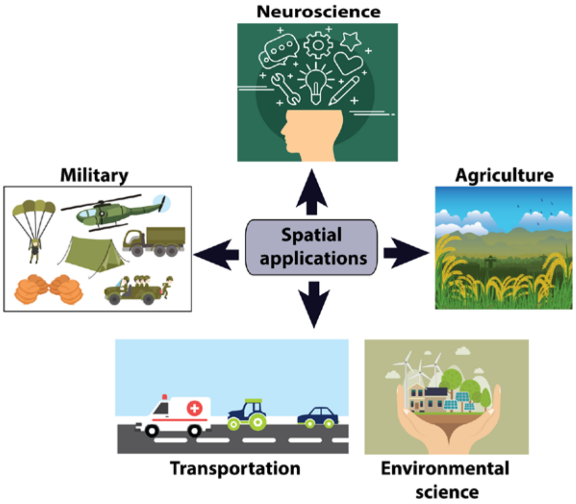

Fig. 3. Spatial applications.

### 2.1. Spatial applications

A vast amount of spatiotemporal data is collected in various applications in the domain of transportation, neuroscience, agriculture, military, healthcare, and environmental science. The motivation for analysis of data involving moving objects in various application domains have been described in this section. Figure 3 illustrates some common spatial applications.

*Transportation*: Taxi pick and drop applications have been used increasingly across the world. Nearby taxis can be found by employing spatial query processing techniques.

*Neuroscience*: Temporal neural activities occurring in brain can be analyzed to understand the principles of brain and identify the mental disorders that made disruption to the normal working behavior of brain. This will help in diagnosis of various neural problems and devise therapeutic plans for patients.

*Agriculture*: Remote-sensing images of agricultural area taken at regular timestamps can help in analyzing the relationships between various factors such as application of fertilizer, planting procedure, and quantity of weeds. This will help to devise strategies to increase crop yield.

*Military*: In the domain of military, there are many applications involving spatiotemporal data. One such application is to find the shortest route for supplying food and weapons to the militants. *Health care*. Temporal data related to diagnosis made to patients and environmental features can be analyzed to discover patterns and spread of epidemic. Effective strategies can be developed for the well-being of humans.

*Environmental science*: The spatiotemporal data regarding atmosphere such as pressure, humidity, and temperature can be analyzed to identify important patterns and devise mitigation strategies for adverse scenarios that might occur in future. For example, the prediction about flow of wind or tornado can be made before the calamity happens in actual. In addition to this, various sensors can be employed in air and water that record data related to particles, ozone, oxygen, and carbon dioxide. This data can be utilized in measuring the quality of air and water in terms of level of pollution calculated using the data from sensors.

### 2.2. *Spatial query processing and indexing*

Spatial point is represented with the help of two coordinates, longitude and latitude. Spatiotemporal data is a spatial data with an additional temporal attribute to define the evolution of object over some period of time. Spatial query extracts the information about the spatial objects present in our environment. Broadly, we can group the spatial queries into three categories, as illustrated in Table 1 that briefly describes each category along with an example query.

Table 1
Categories of spatial queries

| | Description | Example |
|---|---|---|
| Distance query | Deals with distances among spatial objects and their geometries. For example, inside, overlap, encloses, disjoint operations. | Do the two lakes A and B overlap with each other? |
| Direction query | Deals with orientation of spatial objects. For example, top, bottom, left, right operations. | Which houses are north of 30 degrees longitude? |
| Topological query | Deals with relationship between geometries of spatial objects. For example, range, nearest neighbor queries. | Find the three most nearest hospitals to patient's location. |

The structure of spatial data makes it difficult to be processed by traditional data processing methods. Traditional data model such as relational data model suits in scenarios when the schema and structure of data is fixed and the data is one-dimensional. However, spatiotemporal data is multi-dimensional in nature that makes it difficult to apply traditional techniques to process it. This is because the structure and nature of spatiotemporal data introduce a number of challenges from processing and management perspective. One such challenge is reduction of continuous updates.

The eminence of big spatial data has made the management and processing of spatial data, a pervasive component for research. The massive data generated by majority of the GIS applications need an efficient spatial query techniques to keep the response time low. Moreover, the overhead of huge number of update operations performed to track the up to date spatial location of moving objects must be decreased.

Indexing methods aim at minimizing the cost of searching large number of spatial objects and improving the response time of queries. Traditionally, indexing has been used to deal with one-dimensional data. These one-dimensional indices cannot be used in spatial data processing. This is because the structure and nature of spatial data introduce a number of challenges from processing and management perspective. As the data related to moving objects has multiple dimensions, therefore typical one-dimensional indices that are created based on a single dimensional value such as 'salary', cannot handle multi-dimensional values. Therefore, to deal with spatial data, indices must be designed considering the multi-dimensional nature of spatial data. Further, unlike static spatial data, the management and processing of data with dynamic nature (moving objects) needs special consideration. As the moving objects continuously change their position at each time instant, therefore constant updates to indexing structure are required that highly degrades the performance of an application.

### 2.3. Categorization of spatial indices

Primarily, there are two sub branches of spatial indexing methods: space driven and data driven. In space driven structures, spatial objects present in a 2D space are divided into a rectangular grid of cells. Geometric criterion is used as a basis for mapping the spatial objects to cells. Fixed grid had been proposed as a basic space driven structure. It partitions the space into a fixed number of equal sized cells with each cell depicting a secondary storage block. This structure cannot handle skewed data well due to its static data distribution. Therefore, many variations of space driven structures have been proposed to overcome the limitations of basic fixed grid structure. Quadtree, grid file, k-d tree, and various space filling curve techniques have been proposed such as z-order curve, Peano curve, and Hilbert curve. Space filling curves transform a spatial proximity problem from a higher dimensional space to one dimensional space. In contrast to space driven structures, data driven structures are focused on the distribution of spatial objects rather than partitioning of space. R-tree is a data driven structure proposed to handle the indexing of multi-dimensional data. It is a tree based structure that works on the concept of minimum bounding rectangles (MBRs) that hold a group of spatial objects. Leaf node of a tree holds MBR and an object's identifier. It has several limitations such as high overlap among MBRs resulting in high response time of a query. To address these limitations, several variations of R-tree have been proposed in the literature such as R+ Tree, parametric R-tree, FNR R-tree, and X-tree.

With regard to timeline, the indexing of spatiotemporal data can be categorized into three groups. These are: indexing with past spatial locations, indexing with current spatial locations and indexing with future spatial locations. Brief description and distribution of literature with regard to this grouping is defined in Table 2.

Table 2

Categories of spatial indices

| | State-of-the-art indices |
|---|---|
| Past spatial locations | MV3R-tree [206], Muti-level index [29], Unified spatiotemporal index [209], Trajectory-Bundle tree and Spatiotemporal R-tree [163], Fixed Network R-tree [53], MON-tree [37], Historical R-trees [207], Robust index for spatiotemporal archive [64], PA-tree [143], Graph Strip Tree [105], Compressed Start-End Tree [218] |
| Current spatial locations | LU-Grid [231], RUM-tree [188], Fast indexing scheme [94] |
| Future spatial locations | Short lived throw away index [41], Dynamic external memory based index [98], Native space indexing and parametric space indexing [165], Time parameterized R-tree [177], Parametric R-tree [15], Adaptive network R-tree [26] |

### 2.4. Spatial indices for moving objects

The continuously changing data related to moving objects cannot be managed well using traditional database management systems. In traditional relational databases, the data remains unchanged otherwise unless explicitly updated. For example, if the value in employee's age field is 25 then the database will retain this value unless it is explicitly updated. In case of moving objects such as cars, the positional information needs to be updated continuously. Otherwise, this can result in outdated query results. Continuously updating the information results in high performance overhead. Frequent updates cannot be afforded in many real life scenarios. For example, in wireless networks domain, excessive bandwidth will be consumed due to frequent updates. Tracking large number of objects (e.g. millions of cars in a city) becomes a challenge due to frequent update issue. The indexing of moving objects is more challenging due to the overhead of repeatedly updating the index structure in case of location updates. This must be addressed in the design of index to gain performance benefits. Sampling the position of moving object along its trajectory emerged as a traditional way of representing moving objects in indices. One of the earliest research contributions in this regard is presented in [140]. The sampled object location is indexed using an R-tree index. Following it, a few efforts [163,206] are made to update the existing spatial indices to make them handle moving objects. The sampling methods have an overhead of frequent location updates resulting in an overhead of high workload on the system. To overcome this, significant number of efforts are done that model the moving objects using linear functions of time and velocity [9,34,77,156,190,235], and [98]. In this scenario, location update is required only when the velocity of an object changes. This minimizes the number of updates to some extent. The existing studies focused on making predictions during query pro-

Table 3

Possibilities in movement

| | |
|---|---|
| Moving data objects and Moving query | "police van patrolling to search for near by bus"<br>– Data object (bus)<br>– Query (van) |
| Static data objects and moving query | "Which hotel is closest to a taxi?"<br>– Data object (hotel)<br>– Query (taxi) |
| Moving data objects and Static query | "retrieve all the airplanes that will come within 30 miles of the airport in the next 10 minutes"<br>– Data object (airplanes)<br>– Query (airport) |

cessing suffer from performance and functional deficiencies. Most of the research works [71,205,217,245] have considered movement of objects in Euclidean space. However, in realistic scenarios such as road networks, we have to deal with network distances. Many solutions involve prediction models that need to be trained on massive amount of data in the form of trajectories to produce accurate and realistic predictions [8,8,71,81,89,198,205,217,245], and [208]. But, historical data is not available due to privacy concerns.

*Movement possibilities for indexing moving objects*: The continuous spatial query processing systems use a client-server paradigm, consisting of data and query objects. The query objects initiate the query about data objects. There are three possibilities of movement of objects as illustrated in Table 3. These objects share their updated location and query to the server. The server maintains the moving objects and process the initiated queries.

## 3. Landscape of solutions

Recently, the attention of research community has been directed towards integration of spatiotemporal locations and sensor data. The research works related to Geo-IoT integration can be categorizes into five groups: edge computing, device integration, data acquisition and management, service provider, and application related solutions. Recently, a short survey based on this classification has been presented in [96]. The focus of this paper is primarily on indexing of spatiotemporal data generated by IoT devices. It should be noted that existing surveys that have discussed data management issues in Geo-IoT integration have not considered spatiotemporal query processing and indexing data aspects. Therefore, it is vital to have a study in place that explores this domain and its challenges. In this paper, we extensively reviewed the domain of spatiotemporal data indexing (also known as moving objects indexing, interchangeably).

The problem of spatial query processing and management has gained huge attention in recent times. In this section, an overview of whole literature in the light of spatial query processing and management is presented. Some recent solutions have also been discussed briefly. A qualitative comparison of these solutions with regard to design considerations will be presented in the next section.

A thorough study of domain is carried out. There are approximately up to 250 research works that contributed in spatial indexing. The main challenges identified in spatial querying of moving objects include maintaining consistency, enhancing performance, and reducing updates. These will be discussed in detail later. The research works on spatial indexing (for both static and moving objects) starting from early 90s till to date are explored. During exploration of research works on spatial indexing, we focused on both data processing and management aspects. After detailed study, we identified three important concerns that are relevant in spatial indexing. These are mobility of spatial objects, multi-user environment, and massive data processing. Mobility defines whether the spatial objects are moving or static. In real life applications, both static and moving objects play an important role. Some examples of static and moving objects in real life scenarios are defined in Table 3. Multi-user environment is the second most important concern that must be considered while designing an indexing solution. Real life applications are usually designed not only for one user, but for multiple users. Multiple users might initiate queries related to same data items/objects. Thus, synchronization and scheduling of these concurrent queries must be performed to report reliable results to queries. If the scheduling is not performed then this might result in inconsistent query results. Third important factor is the scale of data to be processed. As discussed in the last section, applications in today's era generate huge volume of data that must be processed quickly. Returning the results in timely manner is crucial for many real life applications. For example, in any disaster management application, the results must be processed and disaster alarms must be disseminated timely. Thus, these three characteristics or concerns are quite important to consider for an indexing solution. We defined four features for review and categorization of indexing solutions proposed in the literature. The research works have been grouped into four

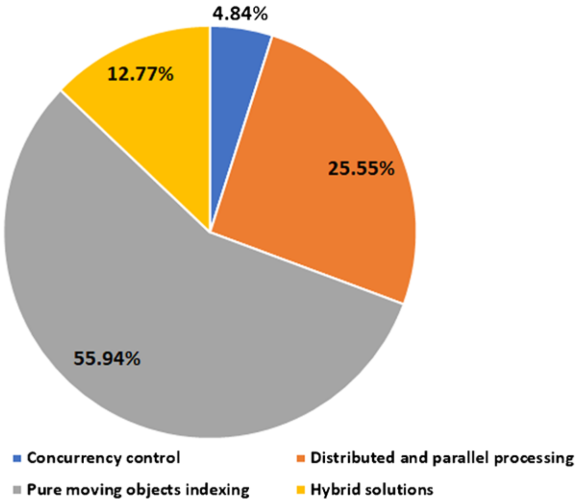

Fig. 4. Categorization of literature.

categories: *Concurrency control*, *distributed and parallel processing*, *pure moving objects indexing*, and *hybrid*. The percentage of research contributions in the literature regarding each category is shown in Fig. 4.

Table 4 provides a comparative view of literature in the light of categories defined.

The *concurrency control* category encloses the research works that have discussed concurrent multiuser queries on spatial data. The *distributed and parallel processing* category discusses the research works that are focused on processing of spatial data and queries in distributed and parallel way. The *pure moving objects indexing* category encloses the research works that are focused primarily on the indexing and query processing of moving objects, i.e. spatiotemporal data. The *hybrid* category encloses the research works that are two or more of the above mentioned categories. Primarily, the works in this category consist of survey and review papers, discussing the domain of spatial query processing generally. The timeline view of these research works with relevant categorization is defined in Table 4. The timeline view of literature is depicted in Fig. 5.

### 3.1. Concurrency control

The paper [88] addressed the issue of maintaining high concurrency in multi-dimensional indices. A new protocol is proposed to eliminate the overhead of blocking during node splits. A top down index region modification approach is proposed that avoids simultaneous locks on nodes at multiple levels. The dataset for experimental evaluation includes multidimensional feature vectors fetched from Corel Image Collection. These feature vectors are indexed in R-tree. The experiments are conducted on shared memory system with multiple processors. Varying workloads for insert and search operations are considered. Insert ratio and disk access ratio is varied and response time of the query is estimated. Scalability is also tested. Experimental results indicate that the proposed scheme provides higher throughput than the state-of-the-art technique, CGist. It also scales well with the increase in number of processors.

Node splits and MBR updates are the two major factors that increase the delay of search operations and degrade the concurrency of indices. To lessen the query delay, an enhanced concurrency control scheme is proposed in [195]. Latches are held only during physical node split. A partial lock coupling method is introduced that enhances concurrency by allowing lock coupling only for shrink MBR operations. MBR shrink operations occur less frequently than expansion operations therefore the proposed solution is more effective in eliminating excessive query delay. For evaluation of proposed method, comparison is made with the existing link technique, CGIST. Dataset with 20 dimensional feature vectors is involved in comparison. Parameters involved in experiments are throughput, response time, varying database and node size, page buffer size, number of dimensions and processes, and the parameter k in kNN queries. The results indicate that the proposed algorithm outperforms the existing algorithms with regard to response time and throughput and also scales well.

The thesis work [35] is focused on the design of an effective concurrency control and indexing method for spatial data. Both stationary and moving objects are put into focus. The main contributions of this research work are: 1) design of an efficient spatial indexing method with object clipping, ZR+-tree. It eliminated the overlap problems in leaf nodes.; 2) concurrency control protocol for object clipping: GLIP; 3) concurrency protocol for linear spatial access indices: CLAM; 4) concurrent query processing for moving spatial objects; 5) Disposable index for Moving Objects (DIME) that eliminates the need for expensive delete operations. In evaluation, compaison of query performance and throughput of ZR+-tree and state-of-the art techniques, R+-tree and R-tree is carried out. To evaluate proposed schemes for moving objects, datasets are generated by Brinkhoff network based moving objects generator. Three range queries related

Table 4

Timeline view of the research articles reviewed

| Year | Concurrency control | Distributed and parallel processing | Pure moving objects indexing | Hybrid |
|---|---|---|---|---|
| 1990 | Lock-based concurrency control technique for spatial index [183] | – | – | Pessimistic and optimistic concurrency control techniques discussion [68] |
| 1995 | Locking technique for R-trees [101] | – | – | – |
| 1997 | – | Parallel algorithm for proximity joins [179] | MOST (Moving objects spatiotemporal) and FTL (Future temporal logic) data models [190] | – |
| 1998 | – | Parallel spatial join [259] | Algorithm for processing FTL in MOST [191] Path planning algorithm in presence of obstacles [54] | – |
| 1999 | Taxonomy of data model, query language, and spatial index [181] | Algorithms for rectangular partitioning [138] Parallel proximity join [3] Parallel spatial join using R-tree [137] Parallel R-tree search algorithm [215] | Discussion of nearest neighbor query in mobile environment [99] Generic index for spatiotemporal data [171] Access structures and their evaluation [141] indices for moving objects [47,98] | – |
| 2000 | Data consistency in spatial databases [261] | Spatial joins [93,113,157] [200] | Data model of moving points and moving regions [62] Data modeling and management survey [221] Moving objects indices [15,163] | Scalable continuous query system [23] |
| 2001 | – | Processing queries in broadcasted spatial indices [66] | Logical concepts and queries in the context of moving objects [197] Model and language for road network [213] Querying mobile objects [165,196,250] [14] | – |
| 2002 | Extensible index [7] Non-blocking index using node-splits [88] | Parallel spatial join algorithm [87] | Indexing trajectories [148,161] Duality transform based indexing [149] Moving objects indexing and querying [2,9,63,72,79,84,103,104,175,176,211], and [167] | – |
| 2003 | – | Distributed index (DIFS) [61] | Range and nearest neighbor queries on moving onjects [1,39,53,76,151,162,164, 169,180,210,233], and [227] | – |
| 2004 | Partial lock coupling [195] | Parallel partitioning and join processing [228] Query routing [262] Virtual memory aggregation technique [60] | Adaptive cell based queries on moving objects [32] Nearest neighbor and range query processing [13,58,85,94,97,111,159,205,224] STRIPES index [156] Modeling moving regions [12] Scalable incremental processing [132] | Query processing and indices [57,77], and [16] |
| 2005 | – | Fully distributed spatial index [108] Load distribution in query processing [109,229] | Moving objects indices for past, current, and future locations [37,73,115,118,139,230,237], and [133] | |
| 2006 | – | Fully distributed spatial index [46] Distributed spatial clustering Route determination using trajectories [128] | [20,130] Indexing past and future locations [25,160] Update tolerant grid based index [231] Motion adaptive index [59] Robust B+tree based index [80] Skyline query processing [75] Reverse nearest neighbor query processing [10] | [136] Cluster based algorithm fr spatial query processing [142] Distributed range query [217] |
| 2007 | – | Hilbert curve based data partitioning [129] Distributed quadtree based indexing [204] | Indexing future trajectories [26] Geometric and probabilistic approach for indexing [192] Exclusive closest pairs in query processing [126] Distributed and continuous query processing [106,150] | [121,225] |

Table 4

(Continued)

| Year | Concurrency control | Distributed and parallel processing | Pure moving objects indexing | Hybrid |
|---|---|---|---|---|
| 2008 | Concurrency control protocol for clipping index [120] | Grid based spatial index [48] | Spatiotemporal indices and query processing algorithms [40,49,235,246], and [30] | Reducing communication cost in distributed query processing [116] Online scalable query execution [131] |
| 2009 | – | Parallel spatial join [248] Distributed index for error prone data broadcast [251] | Euclidean restriction based nearest neighboring algorithm [38] Throw away index [41] Indexing trajectories and locations [24,74,78,158,186,212,244], and [135] | Concurrent operations in spatial query processing [35] |
| 2010 | – | Parallel index based of GPGPU [243] Index using Hadoop based file system [112] | Trajectory and location based query processing and indexing [19] | [155,199] |
| 2011 | – | – | Spatial keyword based query processing [222] Thread level indexing [185] Trajectory based nearest neighbor query [201] Probabilistic count queries [51] Shooting of index images [42] Moving objects query processing [216] | [117] |
| 2012 | – | CudaGIS [241] Parallel kNN join [239] Distributed framework for query processing [189] Spatial partitioning [52,82,145] | Spatial keyword queries [18,172] Intersection join queries [247] Predictive and partial history based querying [50] Moving objects querying algorithm and index [4] | [256] |
| 2013 | – | Parallel spatial mashup [240] Quadtree construction using GPGPU [242] | Activity trajectories [253] Spatial keyword based queries [28] Moving objects indices [178] [5] | – |
| 2014 | – | Large scale parallel query processing [122] Skew resistant spatial join [170] Data partitioning framework [214] Data access methods for spatial query processing [152] | – | Trajectories based indexing [90] |
| 2015 | – | Bitmap indexing [184] Polygon overlay processing [168] Spatial Hadoop [44] | Keyword query processing [27,56] [220,254] Reverse nearest neighbor queries [22] Predictive index [70] | Scalable spatial search [153,238] |
| 2016 | Concurrent Quadtree [258] | Data management on distributed and parallel platforms [236] In memory data management sing locationSpark [202] Spatial query based virtual reality analysis platforms [219] | Keyword aware kNN query [252] In-memory kNN query [17] | – |
| 2017 | Concurrent spatial operations [36] | Big spatial data processing framework [6,65] selectivity estimation [21] Parallel map projection [203] Spatial index for cloud [86] Cumulative sum algorithm [127] | kNN search with road network constraints [182] Neural network based algorithm for spatial predictions [223] Indexing trajectories [232] | Constellation query processing [91] Speculative real time concurrency algorithm [67] |
| 2018 | GPU based index [146] | Parallel algorithm involving polygon intersections [257] Point pattern search [166] Priority R-tree algorithm [43] Index for timeline similarity [69] RDF query processing [144] Batch processing of spatial queries [33] | Throughput optimized adaptive index [124] GPU accelerated index [110] Road aware moving objects mapping [249] Case study on spatial database techniques [134] Range queries [31,173,174,260], and [123] | [55,234], and [147] |

to bicycles, vehicles, and pedestrians are defined for moving objects. Throughput and mobility is evaluated. Correctness analysis of concurrency control protocol is also performed. Findings indicate that proposed protocols provide better results as compared to existing solutions.

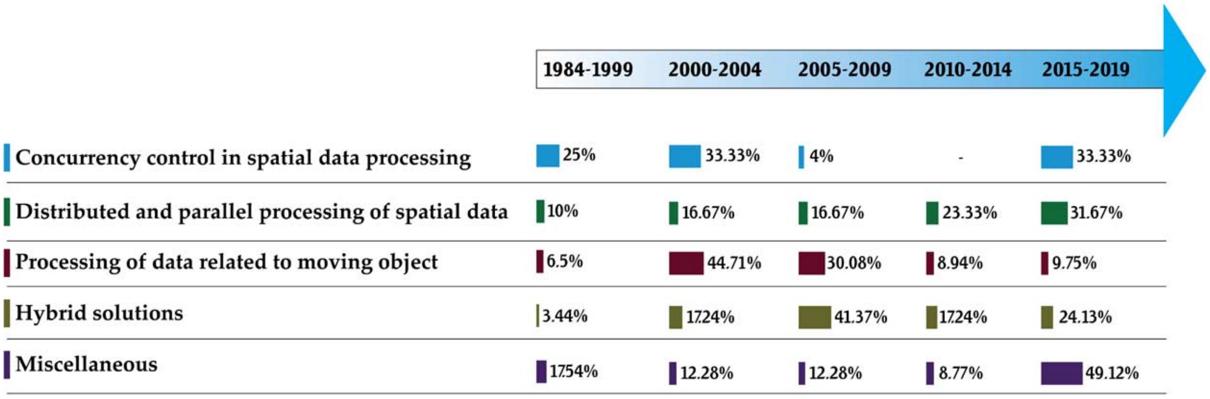

Fig. 5. Timeline view of the research articles reviewed.

In [101], a concurrency control mechanism for R-tree is proposed that is capable to store multi-dimensional spatial data. R-link tree (a variant of R-tree) is implemented and its comparison is performed with R-tree. Illustra database engine is deployed with multiple users sending requests concurrently to a database server. Each client process has its own server in Illustra architecture. To evaluate the performance of proposed protocol, two dimensional dataset with 30600 non-overlapping rectangles is considered. Throughput and response time for insert and search operation is evaluated that indicates that the proposed solution outperforms the existing R-tree index.

In [258], a non-blocking scalable concurrent spatial index is proposed. A decoupling approach for structural adjustment is presented that isolates logical removal and physical adjustment of nodes. The protocol is theoretically evaluated using lemmas and correctness proofs. Experiments are conducted on a single machine with hyper threading. State-of-the-art solutions: ctrie, kary, and patricia are involved in comparative analysis. Two dimensional and uniformly distributed point objects dataset is used in experimentation. Performance in terms of throughput is evaluated for both high and low contention scenarios. The results indicate that the proposed index is better in terms of throughput than the existing solutions.

### 3.2. *Distributed and parallel processing*

In [95], a novel GPU and tree based parallel index: G-tree is proposed that overcome the issue of curse of dimensionality. Existing tree-based methods conventionally use stacks and queues to handle results with GPUs. G-tree uses a novel data structure: structure of arrays for traversals. Breadth first search based traversals are employed instead of depth first search based traversals that resulted in maximization of parallelism over indexing nodes. G-tree use a selective dimensionality filtering approach in which few dimensions out of total number of dimensions are extracted to work on. The extracted set of dimensions are processed on GPUs in a parallel way. The time complexity of a proposed index on range query is theoretically evaluated. Dataset of size up to 9.6 Gb is involved in experiments. Two types of datasets are considered. One dataset is real world dataset involving SURF features with skewed distribution. The other one is a synthetic dataset uniformly generated with dimensionality of up to 256. Experimental results indicate that the performance of the parallel index proposed in this research work is better than the state-of-the-art indexing methods for multi-dimensional data.

The paper [92] proposed a tree based index focussed on both CPU and GPU architectures. CPU is involved in navigation of internal nodes of a tree. Navigation of leaf nodes is performed by GPUs in a linear way utilizing large number of processing units. Host memory of CPU and device memory of GPU is utilized in storing internal and leaf nodes. Various opportunities and guidelines for making balance of the workload between CPU and GPU architectures and co-processing of queries by both the architectures are studied. Three-dimensional point datasets are taken from National Climate Data Center. The datasets have two dimensional longitude and latitude points along with time and number of sensor values such as wind speed, temperature, and precipitation. Various parameters are involved experimental evaluation. These are amount of memory assessed, scan size, average query response time, number of GPU blocks, speed up, average count of conditional instructions with respect to each query, load throughput, number of concurrent queries, query execution time, selection ratio, and number of GPU blocks. The results indicate that the proposed hybrid index provides 12× faster response time for queries as compared to the existing algorithms for traversals.

The paper [116] proposed two different indexing schemes to deal with high communication cost problem in moving object querying. Hybrid distributed solution involving both on-demand and broadcast operations helps in effective querying of moving objects. The server broadcasts the query information repeatedly. Grid of cells are used to represent object movement. During the movement, spatial object downloads the query related information by tuning to broadcast channel without interrupting the server for information. Simulator is designed in Java for evaluating the proposed solution. Parameters varied in simulation are packet size, cell size, and number of objects and queries. Comparison is performed with MobiEyes. The results show that the proposed solution saves 30%–60% of the communication cost.

Two designs of indices: two level hierarchical index and centralized global index are proposed in [139]. To enhance the scalability, replication protocol is proposed for both the index designs. The proposed index extends GMIL to distributed storage. In the proposed scheme, local index is held by each data server. Global index holds MBRS of root nodes and decides about the traversal of local index. To evaluate the two proposed schemes, dataset with three dimensional satellite images of size 30 GB is considered. Storage Resource Broker (SRB) is used to implement distributed indexing schemes. The results indicate that centralized indexing scheme introduces scalability issues for insert and search operations. On the other hand, two level hierarchical index allows better scalability. Replication, however makes centralized scheme faster than the two level hierarchical scheme.

### 3.3. *Pure moving objects indexing*

An efficient range query processing method is proposed in [147] for moving objects. The aim is to minimize the computation and communication cost. Moving objects change their location dynamically, server must be contacted which increases the communication cost. Existing methods use a concept of safe region which ensures that moving object affect the result of query only if it is moving within it. Design of safe region is another challenging task. To avoid this, pruning rules are introduced in this research work. A spatial index: partition retrieved list is proposed that helps in efficiently constructing safe region. Dataset considered for performance evaluation consists of 100K moving objects in LosAngeles. The results show that the computation and communication cost of the proposed solution is low.

The research work [103] proposed an R-tree based index: LUR-tree to deal with moving objects. It reduces unnecessary update operations by ensuring that index gets updated only if the object leaves its corresponding MBR. A sample application of automobile tracking is considered. To evaluate the performance of proposed index, its performance for update and search operation is compared with R$^*$-tree on the basis of number of page accesses. Spatiotemporal data is considered for performance evaluation. The datasets are synthetically generated with uniform and Gaussian distribution. Update query, kNN query, and range search query performance is evaluated. LUR-tree is better in terms of its update performance than R$^*$-tree. Whereas, R$^*$-tree is better with respect to its search performance.

The paper [175] proposed an efficient R$^*$-tree based indexing method to deal with moving objects. With respect to continuous object movement indexing, there are two scenarios. One is to index positions of objects in future and the other one is to index the history of object's movement. Primary focus is on indexing the future movements of objects and making future predictions. Timeslice and window queries are discussed. A skewed 2D data is simulated using a workload generation script Generalized Search Tree Package (GiST). Number of destinations, number of data points, update interval length, querying window size, are query size are considered as workload parameters in simulation.

A hybrid solution is proposed in [117] that can reduce the communication cost incurred during location updates. Two different index structures are proposed, Grid and direction index. Grid of cells is labeled using Hilbert curve. The processing if a query is distributed among moving objects. When a query arrives, the server finds the monitoring region for it. It then updates the relevant data bucket. The server then communicates new monitoring region for this query with the resident objects of an old monitoring region. The proposed index is compared with MobiEyes. The findings indicate that the access time grows linearly with the increase in number of objects.

In [37], a new indexing scheme: MON-tree for moving objects is proposed. Two models of network: edge oriented and route oriented are used. MON-Tree deals with storage and retrieval of past states of moving objects. The index structure is focused on polyline based bounding boxes and movements of objects along them. The bounding boxes make the top of R-tree whereas indexing of movements is performed at the bottom part of the R-tree. The proposed index overcome the problem of high dead space in polyline MBBs. A hash structure is used at the top level of R-tree consisting of identifier for polyline and a pointer for the bottom part of the tree. While inserting an object, polyline identification is used to fetch the bottom R-tree. MBB approximations are utilized in indexing polylines. The bottom part of R-tree stores movement of object represented by a position and time interval. To experimentally evaluate MON-Tree, network based moving objects generator is used consisting of road network of Germany. Dataset contains three models: edge, segment, and route. The comparison is made among the proposed MON-Tree (edge and route model) and FNR-tree. Number of disk accesses are estimated with varying page size, window size, time interval size, speed, number of objects. As per the results, MON-Tree has shown good performance and scalability.

The paper [163] introduces two access structures, TB-tree and STR-tree. The movement of an object is depicted with a polyline representing its trajectory. Two types of spatiotemporal queries are involved in study. These are coordinate and trajectory based queries. The proposed technique overcome the limitation of high overlaps in R-tree. In addition to spatial closeness in R-tree, STR-tree allows trajectory preservation. That is, line segments that belong to same trajectory are kept together. TB-tree preserves trajectories in more strict manner than STR-tree. It bundles up the geometries to trajectory bundle. This trajectory preservance introduces an overhead of minimized spatial discrimination. Combined query processing involving STR-tree and TB-tree is also discussed. Synthetic datasets of line segments are generated using GSTD generator. Comparison of STR-tree and TB-tree is done with R-tree. TB-tree proved out to be more efficient in dealing with trajectory-based queries.

In [226], a tree-based index named as Q+R-tree is proposed for indexing moving objects. It is a combination of Quadtree and R*-tree. The aim is to minimize the cost for update operation. Two types of object movements: fast movement and quasi-static is considered. Quasi-static is defined as a state in which object is neither static nor moving fast. The performance degradation happens mainly due to fast moving objects. Quasi-static and fast moving objects are indexed on R*-tree and Quadtree respectively. Quadtree is better in terms of update performance as compared to R*-tree. On the other hand, R*-tree has better query performance than Quadtree. The combination of these two approaches helps in an efficient indexing. Datasets with moving objects ranging from 100K to 1M are generated for experimentation. Comparison of R-tree, Quadtree, and proposed index is done with respect to insert and search time. Q+R-tree has shown best performance among the rest of the indices.

The paper [77] designed a an indexing scheme: B+tree to index moving objects efficiently. Linearization of moving objects' locations is performed by using space filling curve technique. Subsequently, B+tree indexing is performed. Partitioning of index is done on the basis of update time. Time axis is divided into intervals whose duration is approximated with maximum duration between the two update operations. The update operation is located in a partition defined by a timestamp of a phase during which it occurred. Efficient algorithms for range and KNN queries are defined and discussed. Two versions of index are implemented. One with Z-curve as a space filling technique and other using H-curve. Dataset of moving objects is synthetically generated. Page size, node capacity, maximum update and predictive interval, query window size, dataset size, k for KNN, and number of queries are used as parameters in experimental evaluation. Storage efficiency is also evaluated. The results indicate that proposed index outperformed the rest of the indices. The effects of concurrent access are also estimated. Multi-user environment is simulated by multi-thread programs varying from 1 to 8. Throughput and response time is evaluated. The results indicate that proposed index is more effective as compared to existing index, TPR tree.

In on-demand querying method, client sends their location to server that processes the query and sends the results back to client. If there are large number of simultaneous queries, then service delays are introduced. On-demand environments face a challenge of congestion on server for complicated queries with moving objects. Distributed indices involving client/server model have several limitations such as compromise on privacy, server dependency, and insufficient support for moving objects. In [153], an indexing algorithm is proposed for moving objects in a broadcast environment. It processes queries efficiently by using grid cells. An energy efficient query processing algorithm is also proposed. Cost model is presented and simulation based experiments are performed to evaluate the proposed solution. The comparison is made with Hilbert-curve index. Access and tuning time is involved in comparison. Dataset consisting of 39231 MBRs of Montgomery Country Roads is considered. The results indicate that the proposed algorithm scales well for kNN and range queries and is also energy efficient.

In [17], a scalable kNN query processing technique is proposed that is based on in-memory structure. The focus is on road network with moving objects. R-tree is used to store road network topology. Grid model manages the movement of objects. In kNN, proposed technique: SIMkNN incrementally expands the search region based on distance based evaluation of nearest neighbors. Real and synthetic datasets are involved in experiments. Road network of CA, Los Angles, and USA with 193320 nodes and 264521 edges is taken as real dataset. Moving objects on a network of road are synthetically generated using Brinkhoff road network generator. Overall, simulation of 1 million moving objects is performed. Average performance on random queries is reported. Memory usage, estimation time, I/O count, update time, and response time is evaluated. Results show that SIMkNN performs better in terms of memory and time efficiency as compared with baseline algorithm, MOVNet and IER-PHL.

### 3.4. *Hybrid*

[173] addressed the indexing issue with moving objects for KNN and range queries. The authors proposed a hybrid index based on both grid and R-tree based index structures to provide better performance on search, insert, and update operations. The proposed technique is scalable and robust. R-tree is used at root node and grid based index is used at leaf nodes to aggregate the benefits of the two indexing schemes. R-tree is beneficial for search operations whereas uniform grid based indices are efficient in processing update operations. To experimentally evaluate the effectiveness of the proposed index, comparison is made with R-tree and U-Grid based indices. The comparison is based on the response time of each of the indices with KNN and range based queries. CPU time utilized in update operation is also taken into consideration. Bucket size, node size, grid cell length, number of objects, and percentage of selectivity area is varied and the effect is analyzed. Datasets generated by Open source moving object trace generator (MOTO) are used in experiments. Findings indicate that the proposed solution is scalable with respect to varying number of moving objects.

In [146], authors proposed a GPU based parallel indexing scheme: G-PICS that supports concurrent query processing. The queries are grouped and assigned to a thread. G-PICS supports range query, point query, within-distance query, kNN query, and spatial joins. The performance is evaluated on two types of datasets: real and synthetic with 9.5 million data points each. Real dataset constitutes of molecular simulation of lipid bi-layer system. The data points in real dataset are uniformly distributed whereas synthetic data generated by Zipf distribution is highly skewed. The parameters involved in experimentation include speedup, processing time for range query, and percentage of moving data points. As per the results, the proposed scheme makes good use of resources of GPU by processing large datasets.

The paper [7] proposed an extensible index: SP-GIST supporting space partitioning trees. Space partitioning trees is a class of data structures that hierarchically divide the search space to disjoint portions. With one framework dealing with variety of tree structures is attractive from the database design view point. In this paper [7], common characteristics of spatial space partitioning tree based indices are studied to develop a framework that can represent various tree based structures and eradicate the complexities that hinder the adoption of such tree based structures in database engines. To enhance query performance, interface method named as Cluster is proposed. Three clustering algorithms are presented: fill factor clustering, deep clustering, and breadth clustering. To evaluate the proposed framework, extensions of PR quadtree, MX quadtree, Patricia trie, and trie are implemented. Results indicate that applying clustering techniques help in reducing the number of pages accessed.

In [154], authors proposed a quadtree based index that expresses location of each object as a sequence of bits. It uses a grid based hierarchical division of quadrant. Location of each object is represented with point of interest. Both kNN and range based search algorithms have also been presented. The proposed index is evaluated with both real and synthetic datasets. Real dataset consisting of around 39K data objects represented with MBRs is considered. Access and tuning time is evaluated by varying the size of data. The proposed index is compared with Hilbert curve based distributed spatial index and exponential sequence scheme. It is found that proposed binary index has fastest access time. Hilbert curve based distributed index performed worst among all.

The research works related to moving objects indexing differ from one another on the basis of type of index used (data structure followed in index design), dataset considered (real dataset/simulated workload), and mobility pattern followed. Table 5 provides a comparative view of solutions in the literature regarding moving objects indexing with regard to features: index type, dataset involved in evaluation of proposed index, and movement type supported by each solution.

### 3.5. Discussion

In view of the above discussed literature, we have identified some research gaps and challenges in the design of index for moving objects. The challenges are discussed in detail later in the next section. First, it can be seen that both the spatial and spatiotemporal data have been explored well for indexing. However, these research works have not considered the read/write concurrency conflicts in detail. The effect of multi-user environment and concurrent read/write requests has been studied very briefly. Some research studies that have considered this consistency aspect have used lock based concurrency control schemes, that have an overhead of high latency due to locks. Especially, in case of moving objects, consistency is very critical to maintain because the objects continuously update their locations and proper scheduling of read/write operations is mandatory. Otherwise, inconsistent and outdated results might be reported. Most of the concurrency control protocols in the literature have been designed for static data, typically based on a flat transaction model of a relational database. The protocols proposed for conflict resolution (which is a core component in concurrency control solutions) can be divided into four categories: Pessimistic techniques (2PL) with variants, Optimistic concurrency control techniques with variants, hybrid of 1) and 2), and multi-versioning. Each of these categories has its own pros and cons. Pessimistic techniques have an issue of cascade of blocking and deadlocks. Optimistic concurrency control techniques overcome these issues but have a serious issue of resource wastage. Multi-versioning techniques do not seem fit for long running transactions such as spatial range queries as it results in an overhead of excessive versions. The management of large number of versions is a very challenging task. Therefore, the only feasible solution will be to have a hybrid solution of 1) and 2), covering positive aspects of these techniques. Second, mostly, tree-based indices have been used in which MBRs are used enclose moving objects. Overlapping in these indices has been identified as a very serious issue that can have a negative impact on consistency and performance. Due to overlapping, a single search or update operation might need to traverse multiple nodes of a tree, thus degrading the overall performance. Third, the amount of updates that must be performed as a result of change in object's location must be reduced. In that case, previous obsolete location must be updated with the up to date location. Forth, it is important to note that majority of the contributions in the literature considered only small or medium sized datasets with few million data points only. However, most of the applications in today's era have to deal with huge volume of data. Further, majority of the proposed indices have been evaluated on a single machine architecture. Evaluation in distributed environment is necessary to test scalability of indices. Fifth, in terms of data structure used in indexing, trees and grids have been used in the literature. Both these structures have their own pros and cons. The primary issue with tree based structures is the poor handling of updates. The reasons include expensive operations of split and overlap. Grid based structures, on the other hand, is better in terms of update performance. The idea of integrating these two data structures to implement the benefits of both of these structures has recently got attention in the literature. One such proposal is presented in [173], in which a hybrid index consisting of both tree and grid structures in its design is proposed. Tree based structure helps in achieving good search and query performance whereas grid based structure supports efficient updates. Sixth, the nature of movement considered in moving objects indexing is a very important aspect to consider. Many efficient solutions have been proposed but they only support movement of one of the entities (data object and query object) with the other one assumed as static. These indices are feasible for applications that have either static data objects and moving query objects or static query objects and moving data objects. Sample queries and applications of such scenarios have already been discussed in 3. However, many real life applications have dynamic data objects and query objects. To deal with such applications, solutions proposed in [59,132,135,230], and [175] can be utilized.

Table 5

Comparison of literature on moving objects query processing

| Research work | Year | Spatial index type | Dataset used | Movement type supported | |
|---|---|---|---|---|---|
| | | | | Static data objects/moving query | Moving data objects/moving query |
| [163] | 2000 | tree-based | Synthetic datasets of line segments generated using General Spatio-temporal data (GSTD) generator. | Yes | None |
| [84] | 2002 | grid-based | Synthetic data generated with uniform, skewed, and hyper skewed distribution | Yes | None |
| [103] | 2002 | tree-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [175] | 2008 | tree-based | Simulated objects using a workload generation script Generalized Search Tree Package (GiST) | Yes | Yes |
| [167] | 2002 | tree-based | Synthetic dataset of points | Yes | None |
| [76] | 2003 | tree-based | Simulated objects using a workload generation script Generalized Search Tree Package (GiST) | Yes | None |
| [169] | 2003 | tree-based | Synthetic datasets based on Gauss distribution | None | Yes |
| [226] | 2003 | tree-based | City Simulator, developed at IBM Almaden | Yes | None |
| [132] | 2004 | grid-based | Brinkhoff network based moving objects generator | Yes | Yes |
| [77] | 2004 | tree-based | Synthetic dataset of points | Yes | None |
| [73] | 2005 | tree-based | Simulated dataset using discrete event-driven simulator | Yes | None |
| [133] | 2005 | grid-based | Brinkhoff network based moving objects generator | Yes | None |
| [237] | 2005 | grid-based | Synthetic dataset of points | Yes | None |
| [230] | 2005 | grid-based | Brinkhoff network based moving objects generator | Yes | Yes |
| [37] | 2005 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [136] | 2006 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [75] | 2006 | grid-based | Synthetic datasets involving spatial and non-spatial attributes | Yes | None |
| [10] | 2006 | tree-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [80] | 2006 | tree-based | COST benchmark moving objects generator | Yes | None |
| [59] | 2006 | index structure independent | Synthetic datasets | Yes | Yes |
| [231] | 2006 | grid-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [160] | 2006 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [150] | 2007 | tree-based and grid-based | real spatial dataset of North America | Yes | None |
| [26] | 2007 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [225] | 2007 | grid-based | Brinkhoff network based moving objects generator | Yes | None |
| [246] | 2008 | tree-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [131] | 2008 | grid-based | Brinkhoff network based moving objects generator | Yes | None |

Table 5
(Continued)

| Research work | Year | Spatial index type | Dataset used | Movement type supported | |
|---|---|---|---|---|---|
| | | | | Static data objects/moving query | Moving data objects/moving query |
| [30] | 2008 | tree-based | Synthetic datasets | Yes | None |
| [40] | 2008 | tree-based | Real traffic network data of Beijing | Yes | None |
| [235] | 2008 | tree-based | Synthetic datasets | Yes | None |
| [49] | 2008 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [212] | 2009 | tree-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [24] | 2009 | tree-based | 1) Cellular automation (CA) simulator and 2) Brinkhoff network based moving objects generator | Yes | None |
| [186] | 2009 | both tree-based and grid-based | COST benchmark workloads | Yes | None |
| [244] | 2009 | tree-based | Synthetic datasets | Yes | None |
| [74] | 2009 | algorithm proposed | Synthetic datasets | Yes | None |
| [135] | 2009 | grid-based | Synthetic datasets | Yes | Yes |
| [35] | 2009 | tree-based | 1) Real dataset of California road network and 2) Brinkhoff network generator | Yes | None |
| [19] | 2010 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [216] | 2011 | tree-based | Brinkhoff network based moving objects generator | Yes | None |
| [247] | 2012 | tree-based | Synthetic datasets of points generated using GSTD | Yes | None |
| [5] | 2013 | tree-based | Synthetic datasets | Yes | None |
| [238] | 2015 | grid-based | Synthetic datasets | Yes | None |
| [70] | 2015 | tree-based | Minnesota traffic generator | Yes | None |
| [232] | 2018 | tree-based | Trajectory data from express road system | Yes | None |
| [174] | 2018 | grid-based | 1) Check-in data from Yelp about Los Angeles and 2) Brinkhoff network based moving objects generator | Yes | None |
| [260] | 2018 | tree-based | Real and synthetic dataset from "http://www.chorochronos.org" | Yes | None |
| [173] | 2018 | hybrid index | MOTO (Moving Object Trace generator) | Yes | None |
| [147] | 2018 | grid index | dataset (keyword-based) from Yelp about Los Angeles | Yes | None |
| [31] | 2018 | tree-based | Trajectory data from GeoLife Project | Yes | None |

## 4. Challenges

Overall, five major challenges in continuous spatial indexing are highlighted from the existing literature. These are 1) maintaining consistency, 2) eliminating phantom updates, 3) enhancing performance, 4) avoiding overlap issue, and 5) reduction of updates. These are discussed as under:

### 4.1. Maintaining consistency

There are multiple isolation levels in data management systems to ensure isolated execution of simultaneous read/write. Serializable isolation level is the most protective of all that provides high protection for concurrent queries on one-dimensional data, but they are not optimal for spatial indexing due to underlying expensive locking techniques. Lock based concurrency methods end up in either locking the resource for long,

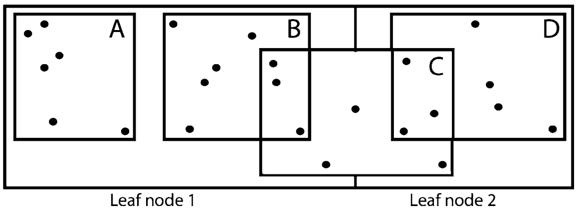

Fig. 6. Lock-based concurrency control in R-tree index.

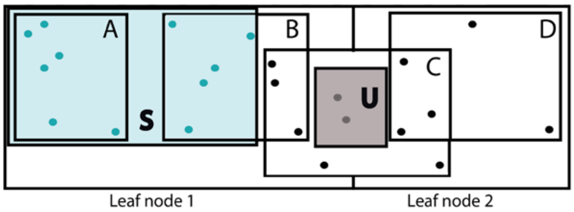

Fig. 7. Inconsistent results due to phantom update issue.

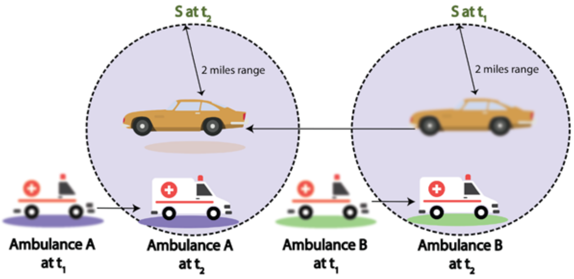

Fig. 8. Phantom update issue in querying of moving objects.

or locking more resources than necessary. The lockable granules have a high overhead due to maintenance of locks. The lock duration must be minimized to improve the throughput. For example, consider the Fig. 6 in which objects are indexed by R-tree index. One solution to ensure consistent results would be to have the search and update windows place shared locks on leaf nodes, A and B. As both the leaf nodes share the object C, therefore another feasible solution would be to have object level locking. This could result in huge number of locks in case of high volume of data. For example, if the search query has to return 1 billion objects as a result, then it means 1 billion locks will be placed at the search time. Thus, no efficient concurrency control techniques exists to ensure serializable operations in moving objects' indexing.

### 4.2. *Eliminating phantom updates*

Phantom update is defined as an update operation occurring before the search operation commits, thereby returning inconsistent results. Consider an example of R-tree index in Fig. 7. A delete operation is performed on data points covered by object C as specified by update window, U. Objects that should be returned by search query window, S are C and E. However, as the delete is performed before the search commits, the deleted object C is also returned as a result. This update/delete operation is referred to as phantom update. One solution to this problem is to lock the area spanned by a update window U to stop search window from accessing the deleted object. In-existence of proper concurrency control measures causes incorrect results returned by a query due to phantom update issue. Figure 8 considers a scenario in which inconsistent results are returned by the query due to lack of concurrency control mechanisms in place. A vehicle S has to keep track of all the ambulances that are traveling within 2 miles range. The query is evaluated at two consecutive timestamps $t_1$ and $t_2$. Two ambulances (ambulance A and ambulance B) are initially traveling outside the query range of S. Ambulance A must be returned as a result at timestamp $t_2$ if everything goes fine. However, if there are no concurrency control mechanisms adopted then this will lead towards following scenarios exhibiting inconsistent results.

- Pseudo disappearance: The query returns null result if it is evaluated before ambulance A updates its new location after deleting previous one.
- Back order: Back order is an issue in which obsolete results are returned by the query. In this case, ambulance B or both ambulances A and B will be returned as an answer to the query.

These inconsistencies can be avoided if a well-designed concurrency control protocol is employed in query processing engine.

### 4.3. *Enhancing performance*

Executing the results of a query in a parallel way can certainly result into high performance gain. However, lock-based concurrency control algorithms introduces difficulties to introduce parallelism in query solution. The queries and location updates occurring at the same time on the shared data item can give inconsistent results. With lock based concurrency control mechanisms, query has to wait for the update operation that holds an exclusive lock. This highly deteriorates the parallelism. Moreover, a long running query on one core introduces long idle period due to locks despite the available resources on other cores of a CPU. As a result of this, workloads comprising of queries and updates cannot scale well as the cores of a CPU will be under-utilized.

### 4.4. *Avoiding overlap issue*

Overlapping is major concern in evaluating a query performance in multi-dimensional index trees. Due to overlapping, a point query might need to go through multiple index tree's branches if the point exists in an overlapped area.

### 4.5. *Reduction of updates*

Unlike static spatial objects, moving objects must have to update their old location with the new one. As the moving objects continuously change their position at each time instant, therefore constant updates are required to indexing structures that highly degrades the performance of an application. To overcome this challenge, pruning rules and effective algorithms must be designed that reduces the number of positional updates relevant to a moving object.

## 5. Design considerations

In this section, some important considerations that are relevant in the design of spatial index are discussed. The comparative view of some recent and potential research works related to indexing of spatial objects is provided in Table 6. The comparison is made with respect to some important design considerations. These design consideration are identified by reviewing the literature and assessing the important aspects related to management and processing of spatial data. We also identified the aspects that are crucial to the workflow of spatial query processing. After thorough analysis, we identified some spatial index design considerations. These design considerations have been discussed in the next sub sections. The motivation for considering these design features is also discussed along with. The relevant research works are also cited that discuss the respective design feature and its significance. This weighs the reliability of our feature selection and justifies the need for considering these design features during spatial index design. The comparative review helps in identification of research gaps for doing research in the domain of spatial indexing.

### 5.1. *Concurrency control protocol used*

As previously discussed, the read/write conflicts must be resolved during spatial query processing to obtain consistent results. The spatial index must have concurrency control protocol embedded to ensure consistency. The core aspect in concurrency control protocol's design is conflict resolution. It can be divided into four categories: Pessimistic techniques (2PL) with variants, Optimistic concurrency control techniques with variants, hybrid of 1) and 2), and multi-versioning. Pessimistic techniques have an issue of cascade of blocking and deadlocks. Optimistic concurrency control techniques overcome these issues but have a serious issue of resource wastage. Multi-versioning techniques do not seem fit for long running transactions such as spatial range queries as it results in an overhead of excessive versions. The management of large number of versions is a very challenging task. Therefore, the only feasible solution will be to have a hybrid solution of 1) and 2), covering positive aspects of these techniques. The need and significance of concurrency control measures for spatial indexing has been discussed in [35] in which phantom update issue, occurring due to simultaneous query processing has been explained. A concurrency control method is proposed for R-tree based spatial index..

### 5.2. *Read/write conflicts*

When multiple spatial queries run, there is a chance of conflict among them. The conflict is resolved using various schemes such as blocking or use of time stamps. It is also important to note whether the conflicting transaction (or query operation) is read or write. This is closely related to the semantics of query results and their consistency. Handling of conflicts among two read queries is different and more critical as compared with that of read write query pair. Because, the updates made by the write operation must be made visible to the simultaneous read operation to ensure consistent results. These conflicts must be minimized in the designed spatial index.

### 5.3. *Spatial query operation*

The multidimensional spatial queries are more complex to handle as compared to one dimensional queries. As discussed earlier, there are three categories of spatial query operations: distance, direction, and topological. Various spatial querying methods specific to

Table 6

Comparison with regard to index design considerations

| Research work | Year | Spatial index type | Geometry | Concurrency control protocol used | Read/write conflicts | Hardware setup | Index design characteristics | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Scalable | Skewness | Overlap | Parallelism |
| [154] | 2019 | grid-based | Moving, Points | Points | None | Single machine | None | None | None | None |
| [95] | 2018 | tree-based | Static, Image dataset feature points | None | None | Single machine | Yes, in terms of dimensionality | Yes | None | Yes |
| [173] | 2018 | hybrid index | Points | None | None | Single machine | Yes | None | None | None |
| [92] | 2018 | tree-based | Static, Points | concurrently navigate the internal tree nodes | Read/write conflicts | Single machine | None | None | None | Yes |
| [146] | 2018 | tree-based | Moving, Points | Parallelize multiple queries running concurrently | Read/write conflicts | Single machine | None | Yes | None | Yes |
| [17] | 2018 | hybrid index | Moving, Points | None | None | Single machine | Yes | Yes | None | Yes |
| [83] | 2018 | grid-based | Static, Polygons | None | None | Single machine | Yes | None | None | Yes |
| [119] | 1997 | tree-based | Static, Points | Lock-based | Read/write conflicts | None, theoretical discussion | None | None | None | None |
| [7] | 2001 | tree-based | Static, Points | Lock-based | Read/write conflicts | None, theoretical discussion | None | None | None | None |
| [88] | 1998 | tree-based | Static, image data | Non-locking, region modification | Read/write conflicts | Single machine | Yes | None | None | None |
| [102] | 2002 | tree-based | Static, Points and polygons | None | None | None, theoretical discussion | None | None | None | None |
| [139] | 2005 | tree-based | Static, image data | None | None | Distributed setup (cluster) | Yes | None | None | Yes |
| [35] | 2009 | tree-based | Moving, Polygons (Roads)-confirm | lock-coupling-based concurrency control | Read/write conflicts | Single machine | Yes | Yes | Yes | Yes |
| [258] | 2018 | tree-based | Moving, Points | Non-blocking | Read/write conflicts | Single machine | None | None | None | None |
| [101] | 1995 | tree-based | Static, Polygons | Lock-based | Read/write conflicts | Single machine | None | None | None | None |
| [194] | 2001 | tree-based | Static, feature vectors | Enhanced partial lock coupling based concurrency control algorithm | Read/write conflicts | Single machine | Yes | None | None | None |
| [117] | 2011 | grid-based | Moving, Points | None | None | None, theoretical discussion | None | None | None | None |

Table 6

(Continued)

| Research work | Year | Spatial index type | Geometry | Concurrency control protocol used | Read/write conflicts | Hardware setup | Index design characteristics | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Scalable | Skewness | Overlap | Parallelism |
| [61] | 2003 | tree-based | Static, Points | None | None | None, theoretical discussion | None | None | None | None |
| [11] | 1996 | tree-based | Static, Points | None | None | Single machine | None | None | Yes | None |
| [204] | 2007 | tree-based | Static, Polygons | None | None | Single machine | None | None | None | None |
| [153] | 2015 | grid-based | Static, Points | None | None | Single machine | Yes | None | None | None |
| [109] | 2005 | grid-based | Static, Points | None | None | Single machine | None | None | None | None |
| [116] | 2008 | grid-based | Static, Points | None | None | Single machine | None | None | None | None |
| [48] | 2008 | grid-based | Static, Points | None | None | Single machine | None | None | None | None |
| [256] | 2012 | Both tree and grid-based | Static, point, line, polygons | None | None | Distributed setup (cluster) | Yes | None | None | None |
| [52] | 2013 | grid-based | Static, Polygons | None | None | 13 virtual machines used for scaling experiments | Yes | None | None | Yes |
| [86] | 2017 | Both tree and grid-based | Static, Points | None | None | Single machine | Yes | None | None | None |
| [152] | 2014 | tree-based | Static, Points | None | None | Single machine | Yes | None | None | None |
| [251] | 2009 | grid-based | Static, Points | None | None | Single machine | Yes | None | None | None |
| [147] | 2018 | Both tree and grid-based | Moving, points | None | None | Single machine | None | Yes | None | None |
| [103] | 2002 | tree-based | Moving, points | None | None | Single machine | None | None | None | None |
| [175] | 2008 | tree-based | Moving, points | None | None | Single machine | Yes | Yes | None | None |
| [37] | 2005 | tree-based | Moving, polylines | None | None | Single machine | Yes | None | None | None |
| [163] | 2000 | tree-based | Moving, lines | None | None | None, theoretical discussion | None | None | None | None |
| [70] | 2015 | tree-based | Moving, points | None | None | Single machine | Yes | None | None | None |
| [227] | 2003 | tree-based | Moving, points | None | None | Single machine | None | None | None | None |
| [77] | 2004 | tree-based | Moving, points | None | None | Single machine | Yes | None | None | None |
| [167] | 2002 | tree-based | Moving, points | None | None | Single machine | Yes | None | None | None |

these query categories have been proposed in the literature. Various indices have also been designed keeping in view the particular spatial query operation such as nearest neighbor query. The spatial index calculates and stores the shortest paths to various data points and query points. It is important to note that calculating these shortest paths is of no use in range based query operation. Thus, the working of spatial index has to be adjusted according to particular query operation under focus. For example, [92,95], and [83] have proposed indices that can handle range queries only. Method proposed in [17] works only for kNN queries. However, in [173], algorithms for both range and kNN queries have been proposed.

### 5.4. Geometry/dimension of data

The core aspect in the design of spatial index is the structure of geometry and dimensions of spatial data with respect to which the spatial index is to be designed. It is highly important to consider this factor as the overall structure and working of spatial index is dependent on the nature of spatial data. For example, the index designed to query data points will not work to query the regions and line-oriented data such as a road path. Thus, it is very important to consider this factor while the design of spatial index. For example, the index proposed in [17] assumes that the data model consists of multi-dimensional points and moving objects do not have any extent. This index cannot at any case store and process moving objects with some extent such as polygons. To deal with polygons we need indices that have been proposed specifically for polygons, e.g. [83].

### 5.5. Nature of movement (data object and query)

For moving data objects, the three possibilities of movements as described earlier are important to consider. One index designed with regard to moving data objects and static query objects cannot be used to support the querying of moving data objects and moving query objects. Thus, while designing a spatial index for an application. All the possible movements must be taken into consideration in index design. The various possibilities of movement of objects along with real life application scenario is illustrated in Table 3. For example, [173] proposed an index for scenario with static data objects and moving query objects. As this index assumes that data objects will not move, therefore location updates for data objects will simply be ignored. This will cause the queries to report inconsistent results with this index. Thus, this index will only work in scenarios where we have static data objects. If the environment has moving data and query objects then indexing solutions that have a support for it must be considered, e.g. [135].

### 5.6. Data model (past trajectories/current locations)

Data model is the essential component in data processing and management. The whole semantic of a spatial query is dependent on the underlying data model. The justification for considering this feature while index design is the same as we discussed for the geometry of data. The extent of spatial object changes the way it is stored and managed by indexing structure. Apart from extent, timeline of queries also heavily affects the design of index. Specifically, in case of query processing with moving objects, the timeline of movement of these objects is very relevant to focus on. For example, if we consider historical data of moving objects, then this data will be in the form of trajectories represented with lines and object identifier. On the other hand, the current location data of moving objects will have longitude and latitude locations with associated timestamp value and object identifier. Thus, the structure of index will highly be dependent on the data model. This must be evaluated while designing of a spatial index.

### 5.7. Scalability

The designed query processing algorithm or index must be able to deal with massive data and scalability demands of an application. The increase in amount of data inserted must not cause the insert operation's performance to degrade. The problem of scalability has been studied with regard to scalable incremental query processing [131,132,230], and [142]. The solutions proposed use a shared execution mechanism or a moving object grouping mechanism to reduce usage of resources. However, the challenge of scaling to multiple servers is not considered. Moreover, the incremental processing seems inappropriate in terms of performance considerations.

### 5.8. Skewness

It is a well-known fact that moving objects have a skewed distribution [107,255], and [193]. If we use a grid based index, then the capacity of cells in a grid structure varies significantly that deteriorates the performance of a query processing algorithm. Consider an example of 4KNN query initiated by query point q. S0 depicts the search space initially explored. This grid encloses only two set of points p1 and p2. The search space needs to be expanded to fetch the other two closes points apart from these two points. When we expand the search space to S1, the network/Euclidean distance must be found from 15 points. Thus, for processing 4KNN query we need a large number of distance calculations which is very expensive computationally. If we can reduce the grid cell size then we will end up in minimizing the number of nearest neighbor calculations. But, this increases the number of grid cells. We have to consider the tradeoff between query performance and storage usage while designing a spatial index.

### 5.9. *Communication cost*

For both tree and grid like structures, there is a communication mechanism that is initiated whenever a merge/split operation occurs on a cell or a node. Whenever a new cell or a node enters or leaves during these operations, the neighborhood must be updated about it. While designing a spatial index, this cost incurred during neighbor communication must be evaluated. Strategies must be designed that can help to improve the performance and minimize the communication cost. It should be noted that communication cost overhead is not only associated with node split or merge operations. The communication aspect goes along all the phases of query processing such as parallel data broadcasting, parallel query processing, intra-cluster communication, grid cells' communication, and communication during location updates. For instance, when the data and query objects move, they have to communicate with the server to send their location updates. Thus, the focus should be to reduce the overhead of communication happening at any phase of spatial query processing. The problem of minimizing the communication cost while spatial query processing has been discussed in [17].

### 5.10. *Overlap*

Overlapping is major concern in evaluating a query performance in multi-dimensional index trees. Due to overlapping, a point query might need to go through multiple index tree's branches if the point exists in an overlapped area. [35] discussed the overlapping issue and proposed a spatial index with zero overlap (ZR-tree).

### 5.11. *Parallelism*

To cater the massive amount of spatial data, there is a need to exploit the modern computational hardware by employing parallelism to increase the performance of a query. Some recent research works, [95,114,185,187], and [48] have discussed the need for parallelism and proposed parallel spatial indexing methods.

## 6. Research opportunities

The domain of spatial data processing can be classified into two groups: modeling and querying. In data modeling sub group, the existing works revolve around proposing a new query language mapped to a real life scenario or designing a better moving object's location prediction function. For example, in [260] range query based on complex shaped obstacles in environment has been proposed. In the second sub group, the focus is primarily on designing an efficient query processing algorithm or index. Some important research opportunities are discussed as under:

- Propose an efficient algorithm or pruning rule (such as safe region techniques) that can cater all the possible movements of data and query objects. This will help in reducing the amount of updates required when the object moves.
- As discussed earlier, the proposed indices are not evaluated with regard to read/write conflicts. It is therefore significant to incorporate read/write conflict resolution mechanisms in the indexing algorithm designed. This can be done by mapping the existing concurrency control solutions in the literature to the spatial indices.
- Majority of the indices designed till date have been evaluated on small datasets on a single machine. However, considering the dynamic nature of spatial data and big data characteristics, it is very important to evaluate these indices for huge data volume in a distributed environment to highlight and test their scalability.
- Parallelism is an effective tool to speedup the response time of any algorithm. At hardware level, parallelism can help to equally utilize all the cores of the system. In the domain of spatial indexing, parallelism can help in meeting the challenges of

high update cost and excessive computations. It would be an interesting work to parallelize the spatial indexing solution and overcome the challenges previously discussed and highlighted in this paper.

## 7. Conclusion

IoT has gained prevalent attention across the globe. It consists of billions of sensors and devices connected through the Internet to acquire, process, manage, and analyze data. Integration of geospatial technology with these sensors has transformed the ways of data processing and management. The use of these sensor nodes and mobile devices has been backed up with issues such as limited network bandwidth and energy consumed by these devices. Thus, the real time spatiotemporal data processing need to use indexing mechanisms to minimize the overhead of extensive query processing. In this paper, we performed a detailed survey of the literature regarding the indexing of real time geospatial data generated by IoT devices. After a thorough review of literature, some major challenges relevant to query processing and indexing of moving objects are highlighted. Various important index design considerations are discussed in detail. The goal is to help researchers in understanding the principles, methods, and challenges in the indexing of real time geospatial data. This will also aid in identifying the future research opportunities.

## References

[1] P.K. Agarwal, L. Arge and J. Erickson, Indexing moving points, *Journal of Computer and System Sciences* **66**(1) (2003), 207–243. doi:10.1016/S0022-0000(02)00035-1.

[2] P.K. Agarwal and C.M. Procopiuc, Advances in indexing for mobile objects, *IEEE Data Eng. Bull.* **25**(2) (2002), 25–34.

[3] R. Agrawal and J.C. Shafer, Method and system for performing proximity joins on high-dimensional data points in parallel, Google Patents, 1999, US Patent 5,884,320.

[4] S. Alamri, D. Taniar and M. Safar, Indexing moving objects in indoor cellular space, in: *2012 15th International Conference on Network-Based Information Systems*, IEEE, 2012, pp. 38–44. doi:10.1109/NBiS.2012.17.

[5] S. Alamri, D. Taniar and M. Safar, Indexing moving objects for directions and velocities queries, *Information Systems Frontiers* **15**(2) (2013), 235–248. doi:10.1007/s10796-012-9367-8.

[6] I.M. Aljawarneh, P. Bellavista, A. Corradi, R. Montanari, L. Foschini and A. Zanotti, Efficient spark-based framework for big geospatial data query processing and analysis, in: *2017 IEEE Symposium on Computers and Communications (ISCC)*, IEEE, 2017, pp. 851–856. doi:10.1109/ISCC.2017.8024633.

[7] W.G. Aref and I.F. Ilyas, An extensible index for spatial databases, in: *Scientific and Statistical Database Management, 2001. SSDBM 2001. Proceedings. Thirteenth International Conference on*, IEEE, 2001, pp. 49–58. doi:10.1109/SSDM.2001.938537.

[8] S. Ayhan, J. Pesce, P. Comitz, D. Sweet, S. Bliesner and G. Gerberick, Predictive analytics with aviation big data, in: *2013 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, IEEE, 2013, pp. 1–13.

[9] R. Benetis, C.S. Jensen, G. Karciauskas and S. Saltenis, Nearest neighbor and reverse nearest neighbor queries for moving objects, in: *Proceedings International Database Engineering and Applications Symposium*, IEEE, 2002, pp. 44–53. doi:10.1109/IDEAS.2002.1029655.

[10] R. Benetis, C.S. Jensen, G. Karĉiauskas and S. Ŝaltenis, Nearest and reverse nearest neighbor queries for moving objects, *The VLDB Journal* **15**(3) (2006), 229–249. doi:10.1007/s00778-005-0166-4.

[11] S. Berchtold, D.A. Keim and H.-P. Kriegel, The X-tree: An index structure for high-dimensional data, in: *Proceedings of the 22th International Conference on Very Large Data Bases, VLDB'96*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996, pp. 28–39, http://dl.acm.org/citation.cfm?id=645922.673502. ISBN 1-55860-382-4.

[12] R. Billen and E. Clementini, A model for ternary projective relations between regions, in: *International Conference on Extending Database Technology*, Springer, 2004, pp. 310–328.

[13] S. Brakatsoulas, D. Pfoser and N. Tryfona, Modeling, storing and mining moving object databases, in: *Proceedings. International Database Engineering and Applications Symposium, 2004. IDEAS'04*, IEEE, 2004, pp. 68–77. doi:10.1109/IDEAS.2004.1319779.

[14] T. Brinkhoff and J. Weitkämper, Continuous queries within an architecture for querying XML-represented moving objects, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 136–154. doi:10.1007/3-540-47724-1_8.

[15] M. Cai and P. Revesz, Parametric R-tree: An index structure for moving objects, in: *Proc. of the COMAD*, 2000.

[16] Y. Cai, K.A. Hua and G. Cao, Processing range-monitoring queries on heterogeneous mobile objects, in: *IEEE International Conference on Mobile Data Management, 2004. Proceedings. 2004*, IEEE, 2004, pp. 27–38.

[17] B. Cao, C. Hou, S. Li, J. Fan, J. Yin, B. Zheng and J. Bao, SIMkNN A Scalable Method for In-Memory kNN Search over Moving Objects in Road Networks, *IEEE Transactions on Knowledge and Data Engineering* (2018).

[18] X. Cao, L. Chen, G. Cong, C.S. Jensen, Q. Qu, A. Skovsgaard, D. Wu and M.L. Yiu, Spatial keyword querying, in: *International Conference on Conceptual Modeling*, Springer, 2012, pp. 16–29. doi:10.1007/978-3-642-34002-4_2.

[19] J.-W. Chang, M.-S. Song and J.-H. Um, TMN-tree: New trajectory index structure for moving objects in spatial networks, in: *2010 10th IEEE International Conference on Computer and Information Technology*, IEEE, 2010, pp. 1633–1638. doi:10.1109/CIT.2010.289.

[20] J.-W. Chang, J.-H. Um and W.-C. LeeP, A new trajectory indexing scheme for moving objects on road networks, in: *British National Conference on Databases*, Springer, 2006, pp. 291–294.

[21] H. Chasparis and A. Eldawy, Experimental evaluation of selectivity estimation on big spatial data, in: *Proceedings of the Fourth International ACM Workshop on Managing and Mining Enriched Geo-Spatial Data*, ACM, 2017, p. 8.

[22] M.A. Cheema, X. Lin, W. Zhang and Y. Zhang, Influence zone: Efficiently processing reverse k nearest neighbors queries, in: *2011 IEEE 27th International Conference on Data Engineering*, IEEE, 2011, pp. 577–588. doi:10.1109/ICDE.2011.5767904.

[23] J. Chen, D.J. DeWitt, F. Tian and Y. Wang, NiagaraCQ: A scalable continuous query system for Internet databases, in: *ACM SIGMOD Record*, Vol. 29, ACM, 2000, pp. 379–390.

[24] J. Chen and X. Meng, Update-efficient indexing of moving objects in road networks, *Geoinformatica* **13**(4) (2009), 397–424. doi:10.1007/s10707-008-0052-5.

[25] J. Chen, X. Meng, Y. Guo, S. Grumbach and H. Sun, Modeling and predicting future trajectories of moving objects in a constrained network, in: *7th International Conference on Mobile Data Management (MDM'06)*, IEEE, 2006, pp. 156–156. doi:10.1109/MDM.2006.107.

[26] J.-D. Chen and X.-F. Meng, Indexing future trajectories of moving objects in a constrained network, *Journal of Computer Science and Technology* **22**(2) (2007), 245–251. doi:10.1007/s11390-007-9031-9.

[27] 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*, IEEE, 2015, pp. 255–266. doi:10.1109/ICDE.2015.7113289.

[28] L. Chen, G. Cong, C.S. Jensen and D. Wu, Spatial keyword query processing: An experimental evaluation, in: *Proceedings of the VLDB Endowment*, Vol. 6, VLDB Endowment, 2013, pp. 217–228.

[29] L. Chen, V. Oria and M.T. Özsu, A multi-level index structure for video databases, in: *Multimedia Information Systems*, 2002, pp. 28–37.

[30] S. Chen, B.C. Ooi, K.-L. Tan and M.A. Nascimento, ST 2 B-tree: A self-tunable spatio-temporal b+-tree index for moving objects, in: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ACM, 2008, pp. 29–42. doi:10.1145/1376616.1376622.

[31] X. Chen, J. Zhang, Z. Xu and J. Liu, HIB-tree: An efficient index method for the big data analytics of large-scale human activity trajectories, *Future Generation Computer Systems* **86** (2018), 1269–1278. doi:10.1016/j.future.2018.01.004.

[32] W. Choi, B. Moon and S. Lee, Adaptive cell-based index for moving objects, *Data & Knowledge Engineering* **48**(1) (2004), 75–101. doi:10.1016/S0169-023X(03)00120-4.

[33] F.M. Choudhury, J.S. Culpepper, Z. Bao and T. Sellis, Batch processing of top-k spatial-textual queries, *ACM Transactions on Spatial Algorithms and Systems (TSAS)* **3**(4) (2018), 13.

[34] A. Civilis, C.S. Jensen, J. Nenortaite and S. Pakalnis, Efficient tracking of moving objects with precision guarantees, in: *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, MOBIQUITOUS 2004, IEEE, 2004, pp. 164–173.

[35] J. Dai, Efficient concurrent operations in spatial databases, PhD thesis, Virginia Tech, 2009.

[36] J.D. Dai and C.-T. Lu, Concurrency control for spatial access, in: *Encyclopedia of GIS*, S. Shekhar, H. Xiong and X. Zhou, eds, Springer International Publishing, Cham, 2017, pp. 285–286. ISBN 978-3-319-17885-1. doi:10.1007/978-3-319-17885-1_175.

[37] V.T. De Almeida and R.H. Güting, Indexing the trajectories of moving objects in networks, *GeoInformatica* **9**(1) (2005), 33–60. doi:10.1007/s10707-004-5621-7.

[38] U. Demiryurek, F. Banaei-Kashani and C. Shahabi, Efficient continuous nearest neighbor query in spatial networks using Euclidean restriction, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2009, pp. 25–43. doi:10.1007/978-3-642-02982-0_5.

[39] R. Ding, X. Meng and Y. Bai, Efficient Index Maintenance for Moving Objects with Future Trajectories, 2003.

[40] Z. Ding, UTR-tree: An index structure for the full uncertain trajectories of network-constrained moving objects, in: *The Ninth International Conference on Mobile Data Management (mdm 2008)*, IEEE, 2008, pp. 33–40. doi:10.1109/MDM.2008.8.

[41] J. Dittrich, L. Blunschi and M.A.V. Salles, Indexing moving objects using short-lived throwaway indexes, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2009, pp. 189–207. doi:10.1007/978-3-642-02982-0_14.

[42] J. Dittrich, L. Blunschi and M.A.V. Salles, MOVIES: Indexing moving objects by shooting index images, *Geoinformatica* **15**(4) (2011), 727–767. doi:10.1007/s10707-011-0122-y.

[43] A. Elashry, A. Shehab, A. Riad and A. Aboul-Fotouh, 2DPR-Tree: Two-dimensional priority r-tree algorithm for spatial partitioning in spatialhadoop, *ISPRS International Journal of Geo-Information* **7**(5) (2018), 179. doi:10.3390/ijgi7050179.

[44] A. Eldawy, L. Alarabi and M.F. Mokbel, Spatial partitioning techniques in SpatialHadoop, *Proceedings of the VLDB Endowment* **8**(12) (2015), 1602–1605. doi:10.14778/2824032.2824057.

[45] A. Eldawy and M.F. Mokbel, The era of big spatial data, in: *2015 31st IEEE International Conference on Data Engineering Workshops*, IEEE, 2015, pp. 42–49. doi:10.1109/ICDEW.2015.7129542.

[46] S.H. Eo, S. Pandey, M.-K. Kim, Y.-H. Oh and H.-Y. Bae, FDSI-Tree: A fully distributed spatial index tree for efficient & power-aware range queries in sensor networks, in: *International Conference on Current Trends in Theory and Practice of Computer Science*, Springer, 2006, pp. 254–261.

[47] M. Erwig, R.H. Gu, M. Schneider, M. Vazirgiannis et al., Spatio-temporal data types: An approach to modeling and querying moving objects in databases, *GeoInformatica* **3**(3) (1999), 269–296. doi:10.1023/A:1009805532638.

[48] M. Esch, J. Botev, H. Schloss and I. Scholtes, Gp3-a distributed grid-based spatial index infrastructure for massive multiuser virtual environments, in: *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*, IEEE, 2008, pp. 811–816. doi:10.1109/ICPADS.2008.13.

[49] Y. Fang, J. Cao, Y. Peng and L. Wang, Indexing the past, present and future positions of moving objects on fixed networks, in: *2008 International Conference on Computer Science and Software Engineering*, Vol. 4, IEEE, 2008, pp. 524–527. doi:10.1109/CSSE.2008.1449.

[50] Y. Fang, J. Cao, J. Wang, Y. Peng and W. Song, HTPR*-Tree: An efficient index for moving objects to support predictive query and partial history query, in: *International Conference on Web-Age Information Management*, Springer, 2011, pp. 26–39.

[51] A. Follmann, M.A. Nascimento, A. Züfle, M. Renz, P. Kröger and H.-P. Kriegel, Continuous probabilistic count queries in wireless sensor networks, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2011, pp. 279–296. doi:10.1007/978-3-642-22922-0_17.

[52] A. Fox, C. Eichelberger, J. Hughes and S. Lyon, Spatio-temporal indexing in non-relational distributed databases, in: *Big Data, 2013 IEEE International Conference on*, IEEE, 2013, pp. 291–299. doi:10.1109/BigData.2013.6691586.

[53] E. Frentzos, Indexing objects moving on fixed networks, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2003, pp. 289–305. doi:10.1007/978-3-540-45072-6_17.

[54] K. Fujimura and H. Samet, Path planning among moving obstacles using spatial indexing, in: *ICRA*, 1988, pp. 1662–1667.

[55] N. Gao, A Paradigm for Scalable, Transactional, and Efficient Spatial Indexes, PhD thesis, University of Colorado at Boulder, 2018.

[56] Y. Gao, J. Zhao, B. Zheng and G. Chen, Efficient collective spatial keyword query processing on road networks, *IEEE Transactions on Intelligent Transportation Systems* **17**(2) (2015), 469–480. doi:10.1109/TITS.2015.2477837.

[57] B. Gedik and L. Liu, Mobieyes: Distributed processing of continuously moving queries on moving objects in a mobile system, in: *International Conference on Extending Database Technology*, Springer, 2004, pp. 67–87.

[58] B. Gedik, K.-L. Wu, P. Yu and L. Liu, Motion adaptive indexing for moving continual queries over moving objects, in: *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, ACM, 2004, pp. 427–436.

[59] B. Gedik, K.-L. Wu, P.S. Yu and L. Liu, Processing moving queries over moving objects using motion-adaptive indexes, *IEEE Transactions on Knowledge and Data Engineering* **18**(5) (2006), 651–668. doi:10.1109/TKDE.2006.81.

[60] M. Gorawski and R. Malczok, Distributed spatial data warehouse indexed with virtual memory aggregation tree, in: *STDBM*, 2004, pp. 25–32.

[61] B. Greenstein, S. Ratnasamy, S. Shenker, R. Govindan and D. Estrin, DIFS: A distributed index for features in sensor networks, *Ad Hoc Networks* **1**(2–3) (2003), 333–349. doi:10.1016/S1570-8705(03)00010-6.

[62] R.H. Güting, M.H. Böhlen, M. Erwig, C.S. Jensen, N.A. Lorentzos, M. Schneider and M. Vazirgiannis, A foundation for representing and querying moving objects, *ACM Transactions on Database Systems (TODS)* **25**(1) (2000), 1–42. doi:10.1145/352958.352963.

[63] M. Hadjieleftheriou, G. Kollios, V.J. Tsotras and D. Gunopulos, Efficient indexing of spatiotemporal objects, in: *International Conference on Extending Database Technology*, Springer, 2002, pp. 251–268.

[64] M. Hadjieleftheriou, G. Kollios, V.J. Tsotras and D. Gunopulos, Indexing spatiotemporal archives, *The VLDB Journal* **15**(2) (2006), 143–164. doi:10.1007/s00778-004-0151-3.

[65] S. Hagedorn, P. Götze and K.-U. Sattler, Big spatial data processing frameworks: Feature and performance evaluation, in: *EDBT*, 2017, pp. 490–493.

[66] S. Hambrusch, C.-M. Liu, W.G. Aref and S. Prabhakar, Query processing in broadcasted spatial index trees, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 502–521. doi:10.1007/3-540-47724-1_26.

[67] S. Hamdi, E. Bouazizi and S. Faiz, A speculative concurrency control in real-time spatial big data using real-time nested spatial transactions and imprecise computation, in: *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, IEEE, 2017, pp. 534–540.

[68] G. Hanssen, Concurrency control in distributed geographical database systems, in: *ScanGIS*, 2003, pp. 167–180.

[69] Z. He and X. Ma, A distributed indexing method for timeline similarity query, *Algorithms* **11**(4) (2018), 41. doi:10.3390/a11040041.

[70] A.M. Hendawi, J. Bao, M.F. Mokbel and M. Ali, Predictive tree: An efficient index for predictive queries on road networks, in: *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, IEEE, 2015, pp. 1215–1226. doi:10.1109/ICDE.2015.7113369.

[71] A.M. Hendawi and M.F. Mokbel, Panda: A predictive spatio-temporal query processor, in: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, ACM, 2012, pp. 13–22.

[72] K. Hornsby and M.J. Egenhofer, Modeling moving objects over multiple granularities, *Annals of Mathematics and Artificial Intelligence* **36**(1–2) (2002), 177–194. doi:10.1023/A:1015812206586.

[73] H. Hu, J. Xu and D.L. Lee, A generic framework for monitoring continuous spatial queries over moving objects, in: *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, ACM, 2005, pp. 479–490. doi:10.1145/1066157.1066212.

[74] Y.-K. Huang, Z.-W. Chen and C. Lee, Continuous k-nearest neighbor query over moving objects in road networks, in: *Advances in Data and Web Management*, Springer, 2009, pp. 27–38. doi:10.1007/978-3-642-00672-2_5.

[75] Z. Huang, H. Lu, B.C. Ooi and A.K. Tung, Continuous skyline queries for moving objects, *IEEE Transactions on Knowledge and Data Engineering* **18**(12) (2006), 1645–1658. doi:10.1109/TKDE.2006.185.

[76] G.S. Iwerks, H. Samet and K. Smith, Continuous k-nearest neighbor queries for continuously moving points with updates, in: *Proceedings of the 29th International Conference on Very Large Data Bases-Volume 29*, VLDB Endowment, 2003, pp. 512–523.

[77] C.S. Jensen, D. Lin and B.C. Ooi, Query and update efficient B+-tree based indexing of moving objects, in: *Proceedings of the Thirtieth International Conference on Very Large Data Bases-Volume 30*, VLDB Endowment, 2004, pp. 768–779.

[78] C.S. Jensen, H. Lu and B. Yang, Indexing the trajectories of moving objects in symbolic indoor space, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2009, pp. 208–227. doi:10.1007/978-3-642-02982-0_15.

[79] C.S. Jensen and S. Saltenis, Towards increasingly update efficient moving-object indexing, *IEEE Data Eng. Bull.* **25**(2) (2002), 35–40.

[80] C.S. Jensen, D. Tielsytye and N. Tradilauskas, Robust B+-tree-based indexing of moving objects, in: *7th International Conference on Mobile Data Management (MDM'06)*, IEEE, 2006, pp. 12–12. doi:10.1109/MDM.2006.135.

[81] H. Jeung, M.L. Yiu, X. Zhou and C.S. Jensen, Path prediction and predictive range querying in road network databases, *The VLDB Journal* **19**(4) (2010), 585–602. doi:10.1007/s00778-010-0181-y.

[82] Y. Ji and N. Geroliminis, On the spatial partitioning of urban transportation networks, *Transportation Research Part B: Methodological* **46**(10) (2012), 1639–1656. doi:10.1016/j.trb.2012.08.005.

[83] H. Jiang, J. Kang, Z. Du, F. Zhang, X. Huang, R. Liu and X. Zhang, Vector spatial big data storage and optimized query based on the multi-level Hilbert grid index in HBase, *Information* **9**(5) (2018), 116. doi:10.3390/info9050116.

[84] D. Kalashnikov, S. Prabhakar, S. Hambrusch and W. Aref, Efficient evaluation of continuous range queries on moving objects, in: *International Conference on Database and Expert Systems Applications*, Springer, 2002, pp. 731–740. doi:10.1007/3-540-46146-9_72.

[85] D.V. Kalashnikov, S. Prabhakar and S.E. Hambrusch, Main memory evaluation of monitoring queries over moving objects, *Distributed and Parallel Databases* **15**(2) (2004), 117–135. doi:10.1023/B:DAPD.0000013068.25976.88.

[86] I. Kamel, A.M. Talha and Z.A. Aghbari, Dynamic spatial index for efficient query processing on the cloud, *Journal of Cloud Computing* **6**(1) (2017), 5. doi:10.1186/s13677-017-0077-0.

[87] M.-S. Kang, S.-K. Ko, K. Koh and Y.-C. Choy, A parallel spatial join processing for distributed spatial databases, in: *International Conference on Flexible Query Answering Systems*, Springer, 2002, pp. 212–225. doi:10.1007/3-540-36109-X_17.

[88] K.R. Kanth, D. Serena and A.K. Singh, Improved concurrency control techniques for multi-dimensional index structures, in: *Parallel Processing Symposium, 1998. IPPS/SPDP 1998*, Proceedings of the First Merged International... and Symposium on Parallel and Distributed Processing 1998, IEEE, 1998, pp. 580–586.

[89] H.A. Karimi and X. Liu, A predictive location model for location-based services, in: *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, ACM, 2003, pp. 126–133.

[90] S. Ke, J. Gong, S. Li, Q. Zhu, X. Liu and Y. Zhang, A hybrid spatio-temporal data indexing method for trajectory databases, *Sensors* **14**(7) (2014), 12990–13005. doi:10.3390/s140712990.

[91] A. Khatibi, F. Porto, J.G. Rittmeyer, E. Ogasawara, P. Valduriez and D. Shasha, Pre-processing and indexing techniques for constellation queries in big data, in: *International Conference on Big Data Analytics and Knowledge Discovery*, Springer, 2017, pp. 164–172. doi:10.1007/978-3-319-64283-3_12.

[92] J. Kim and B. Nam, Co-processing heterogeneous parallel index for multi-dimensional datasets, *Journal of Parallel and Distributed Computing* **113** (2018), 195–203. doi:10.1016/j.jpdc.2017.10.015.

[93] J.-D. Kim and B.-H. Hong, Parallel spatial joins using grid files, in: *Proceedings Seventh International Conference on Parallel and Distributed Systems (Cat. No. PR00568)*, IEEE, 2000, pp. 531–536.

[94] K.-S. Kim, S.-W. Kim, T.-W. Kim and K.-J. Li, Fast indexing and updating method for moving objects on road networks, in: *Fourth International Conference on Web Information Systems Engineering Workshops, 2003. Proceedings*, IEEE, 2003, pp. 34–42.

[95] M. Kim, L. Liu and W. Choi, A GPU-aware parallel index for processing high-dimensional Big Data, *IEEE Transactions on Computers* (2018).

[96] M.-S. Kim, Research issues and challenges related to Geo-IoT platform, *Spatial Information Research* **26**(1) (2018), 113–126. doi:10.1007/s41324-017-0161-z.

[97] M.R. Kolahdouzan and C. Shahabi, Continuous K-nearest neighbor queries in spatial network databases, in: *STDBM*, Citeseer, 2004, pp. 33–40.

[98] G. Kollios, D. Gunopulos and V.J. Tsotras, On indexing mobile objects, in: *Proceedings of the Eighteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM, 1999, pp. 261–272. doi:10.1145/303976.304002.

[99] G. Kollios, D. Gunopulos and V.J. Tsotras, Nearest neighbor queries in a mobile environment, in: *Spatio-Temporal Database Management*, Springer, 1999, pp. 119–134. doi:10.1007/3-540-48344-6_7.

[100] K. Koperski, J. Adhikary and J. Han, Spatial data mining: Progress and challenges survey paper, in: *Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, Citeseer, Montreal, Canada, 1996, pp. 1–10.

[101] M. Kornacker and D. Banks, High-concurrency locking in R-trees, in: *VLDB*, Vol. 95, 1995, pp. 134–145.

[102] R.K.V. Kothuri, S. Ravada and D. Abugov, Quadtree and R-tree indexes in oracle spatial: A comparison using GIS data, in: *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, ACM, 2002, pp. 546–557. doi:10.1145/564691.564755.

[103] D. Kwon, S. Lee and S. Lee, Indexing the current positions of moving objects using the lazy update R-tree, in: *Mobile Data Management*, Proceedings. Third International Conference on, IEEE, 2002, pp. 113–120.

[104] I. Lazaridis, K. Porkaew and S. Mehrotra, Dynamic queries over mobile objects, in: *International Conference on Extending Database Technology*, Springer, 2002, pp. 269–286.

[105] T.T.T. Le and B.G. Nickerson, Efficient search of moving objects on a planar graph, in: *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2008, p. 41.

[106] C.-C. Lee, Y.-H. Wu and A.L. Chen, Continuous evaluation of fastest path queries on road networks, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2007, pp. 20–37. doi:10.1007/978-3-540-73540-3_2.

[107] K.C. Lee, M. Le, J. Harri and M. Gerla, Louvre: Landmark overlays for urban vehicular routing environments, in: *2008 IEEE 68th Vehicular Technology Conference*, IEEE, 2008, pp. 1–5.

[108] W.-C. Lee and B. Zheng, DSI: A fully distributed spatial index for location-based wireless broadcast services, in: *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, IEEE, 2005, pp. 349–358.

[109] W.-C. Lee and B. Zheng, A Fully Distributed Spatial Index for Wireless Data Broadcast, in: *Data Engineering, International Conference on*, IEEE, 2005, pp. 417–418.

[110] C. Li, Y. Gu, J. Qi, J. He, Q. Deng and G. Yu, A gpu accelerated update efficient index for knn queries in road networks, in: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, IEEE, 2018, pp. 881–892. doi:10.1109/ICDE.2018.00084.

[111] Y. Li, J. Yang and J. Han, Continuous k-nearest neighbor search for moving objects, in: *Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004*, IEEE, 2004, pp. 123–126.

[112] H. Liao, J. Han and J. Fang, Multi-dimensional index on hadoop distributed file system, in: *2010 IEEE Fifth International Conference on Networking, Architecture, and Storage*, IEEE, 2010, pp. 240–249. doi:10.1109/NAS.2010.44.

[113] A.A. Lima, C. Esperança and M. Mattoso, A parallel spatial join framework using PMR-quadtrees, in: *Proceedings 11th International Workshop on Database and Expert Systems Applications*, IEEE, 2000, pp. 889–893. doi:10.1109/DEXA.2000.875131.

[114] S.V. Limkar and R.K. Jha, A novel method for parallel indexing of real time geospatial big data generated by IoT devices, *Future Generation Computer Systems* **97** (2019), 433–452. doi:10.1016/j.future.2018.09.061.

[115] D. Lin, C.S. Jensen, B.C. Ooi and S. Šaltenis, Efficient indexing of the historical, present, and future positions of moving objects, in: *Proceedings of the 6th International Conference on Mobile Data Management*, ACM, 2005, pp. 59–66. doi:10.1145/1071246.1071256.

[116] F. Liu, K.A. Hua and F. Xie, On reducing communication cost for distributed moving query monitoring systems, in: *Mobile Data Management, 2008. MDM'08. 9th International Conference on*, IEEE, 2008, pp. 156–164.

[117] F. Liu, K.A. Hua and F. Xie, A hybrid communication solution to distributed moving query monitoring systems, *Electronic Commerce Research and Applications* **10**(2) (2011), 214–226. doi:10.1016/j.elerap.2010.08.007.

[118] Z.-H. Liu, X.-L. Liu, J.-W. Ge and H.-Y. Bae, Indexing large moving objects from past to future with PCFI+-index, in: *COMAD*, 2005, pp. 131–137.

[119] D. Lomet and B. Salzberg, Concurrency and recovery for index trees, *The VLDB Journal – The International Journal on Very Large Data Bases* **6**(3) (1997), 224–240.

[120] C.-T. Lu, J. Dai, Y. Jin and J. Mathuria, GLIP: A concurrency control protocol for clipping indexing, *IEEE Transactions on Knowledge and Data Engineering* **21**(5) (2009), 714–728. doi:10.1109/TKDE.2008.183.

[121] H. Lu, Z. Huang, C.S. Jensen and L. Xu, Distributed, concurrent range monitoring of spatial-network constrained mobile objects, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2007, pp. 403–422. doi:10.1007/978-3-540-73540-3_23.

[122] J. Lu, Parallel SECONDO: Processing moving objects data at large scale, PhD thesis, Fern Universität in Hagen, 2014.

[123] V. Lund, Scalable Region Query Processing in Spatial and Spatiotemporal Databases, Master's thesis, NTNU, 2018.

[124] S. Luo, B. Kao, G. Li, J. Hu, R. Cheng and Y. Zheng, TOAIN: A throughput optimizing adaptive index for answering dynamic k nn queries on road networks, *Proceedings of the VLDB Endowment* **11**(5) (2018), 594–606. doi:10.1145/3187009.3177736.

[125] A.R. Mahmood, S. Punni and W.G. Aref, Spatio-temporal access methods: A survey (2010–2017), *GeoInformatica* **23**(1) (2019), 1–36. doi:10.1007/s10707-018-0329-2.

[126] N. Mamoulis, M.L. Yiu et al., Continuous monitoring of exclusive closest pairs, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2007, pp. 1–19.

[127] G. Manogaran and D. Lopez, Spatial cumulative sum algorithm with big data analytics for climate change detection, *Computers & Electrical Engineering* **65** (2018), 207–221. doi:10.1016/j.compeleceng.2017.04.006.

[128] A. Meka and A.K. Singh, Distributed spatial clustering in sensor networks, in: *International Conference on Extending Database Technology*, Springer, 2006, pp. 980–1000.

[129] L. Meng, C. Huang, C. Zhao and Z. Lin, An improved Hilbert curve for parallel spatial data partitioning, *Geo-spatial Information Science* **10**(4) (2007), 282–286. doi:10.1007/s11806-007-0107-z.

[130] K.-W. Min, J.-W. Kim and J.-H. Park, Optimal route determination technology based on trajectory querying moving object database, in: *International Conference on Database and Expert Systems Applications*, Springer, 2006, pp. 666–675. doi:10.1007/11827405_65.

[131] M.F. Mokbel and W.G. Aref, SOLE: Scalable on-line execution of continuous queries on spatio-temporal data streams, *The VLDB Journal – The International Journal on Very Large Data Bases* **17**(5) (2008), 971–995.

[132] M.F. Mokbel, X. Xiong and W.G. Aref, SINA: Scalable incremental processing of continuous queries in spatio-temporal databases, in: *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, ACM, 2004, pp. 623–634. doi:10.1145/1007568.1007638.

[133] M.F. Mokbel, X. Xiong, M.A. Hammad and W.G. Aref, Continuous query processing of spatio-temporal data streams in place, *GeoInformatica* **9**(4) (2005), 343–365. doi:10.1007/s10707-005-4576-7.

[134] K. Mouratidis, *Applying Spatial Database Techniques to Other Domains: A Case Study on Top-k and Computational Geometric Operators, in: Proceedings of the Fifth International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data*, ACM, 2018, pp. 25–26.

[135] K. Mouratidis, S. Bakiras and D. Papadias, Continuous monitoring of spatial queries in wireless broadcast environments, *IEEE Transactions on Mobile Computing* **8**(10) (2009), 1297–1311. doi:10.1109/TMC.2009.14.

[136] K. Mouratidis, M.L. Yiu, D. Papadias and N. Mamoulis, Continuous nearest neighbor monitoring in road networks, in: *Proceedings of the 32nd International Conference on Very Large Data Bases*, VLDB Endowment, 2006, pp. 43–54.

[137] L. Mutenda and M. Kitsuregawa, Parallel r-tree spatial join for a shared-nothing architecture, in: *Proceedings 1999 International Symposium on Database Applications in Non-Traditional Environments (DANTE'99) (Cat. No. PR00496)*, IEEE, 1999, pp. 423–430.

[138] S. Muthukrishnan, V. Poosala and T. Suel, On rectangular partitionings in two dimensions: Algorithms, complexity and applications, in: *International Conference on Database Theory*, Springer, 1999, pp. 236–256.

[139] B. Nam and A. Sussman, Spatial indexing of distributed multidimensional datasets, in: *Cluster Computing and the Grid*, CCGrid 2005. IEEE International Symposium on, Vol. 2, IEEE, 2005, pp. 743–750.

[140] M.A. Nascimento and J.R. Silva, Towards historical R-trees, in: *Proceedings of the 1998 ACM Symposium on Applied Computing*, ACM, 1998, pp. 235–240. doi:10.1145/330560.330692.

[141] M.A. Nascimento, J.R. Silva and Y. Theodoridis, Evaluation of access structures for discretely moving points, in: *Spatio-Temporal Database Management*, Springer, 1999, pp. 171–189. doi:10.1007/3-540-48344-6_10.

[142] R.V. Nehme and E.A. Rundensteiner, SCUBA: Scalable cluster-based algorithm for evaluating continuous spatio-temporal queries on moving objects, in: *International Conference on Extending Database Technology*, Springer, 2006, pp. 1001–1019.

[143] J. Ni and C.V. Ravishankar, PA-tree: A parametric indexing scheme for spatio-temporal trajectories, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2005, pp. 254–272. doi:10.1007/11535331_15.

[144] P. Nikitopoulos, A. Vlachou, C. Doulkeridis and G.A. Vouros, DiStRDF: Distributed spatio-temporal RDF queries on spark, in: *EDBT/ICDT Workshops*, 2018, pp. 125–132.

[145] E.P. Nora, B.R. Lajoie, E.G. Schulz, L. Giorgetti, I. Okamoto, N. Servant, T. Piolot, N.L. van Berkum, J. Meisig, J. Sedat et al., Spatial partitioning of the regulatory landscape of the X-inactivation centre, *Nature* **485**(7398) (2012), 381. doi:10.1038/nature11049.

[146] Z. Nouri and Y.-C. Tu, GPU-based parallel indexing for concurrent spatial query processing, in: *Proceedings of the 30th International Conference on Scientific and Statistical Database Management*, ACM, 2018, p. 23.

[147] S. Oh, H. Jung and U.-M. Kim, An efficient processing of range spatial keyword queries over moving objects, in: *Information Networking (ICOIN), 2018 International Conference on*, IEEE, 2018, pp. 525–530. doi:10.1109/ICOIN.2018.8343174.

[148] D. Papadias, Y. Tao, J. Zhang, N. Mamoulis, Q. Shen and J. Sun, Indexing and retrieval of historical aggregate information about moving objects, 2002.

[149] D. Papadopoulos, G. Kollios, D. Gunopulos and V.J. Tsotras, Indexing mobile objects using duality transforms, *IEEE Data Eng. Bull.* **25**(2) (2002), 18–24.

[150] S. Papadopoulos, D. Sacharidis and K. Mouratidis, Continuous medoid queries over moving objects, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2007, pp. 38–56. doi:10.1007/978-3-540-73540-3_3.

[151] H.K. Park, J.H. Son and M.H. Kim, An efficient spatiotemporal indexing method for moving objects in mobile communication environments, in: *International Conference on Mobile Data Management*, Springer, 2003, pp. 78–91. doi:10.1007/3-540-36389-0_6.

[152] K. Park, Efficient data access for location-dependent spatial queries, *Journal of Computer Science and Technology* **29**(3) (2014), 449–469. doi:10.1007/s11390-014-1442-9.

[153] K. Park, An efficient scalable spatial data search for location-aware mobile services, *J. Inf. Sci. Eng.* **31**(1) (2015), 165–178.

[154] K. Park, A hierarchical binary quadtree index for spatial queries, *Wireless Networks* **25**(4) (2019), 1913–1929. doi:10.1007/s11276-018-1661-z.

[155] K. Park, H. Choo and P. Valduriez, A scalable energy-efficient continuous nearest neighbor search in wireless broadcast systems, *Wireless Networks* **16**(4) (2010), 1011–1031. doi:10.1007/s11276-009-0185-y.

[156] J.M. Patel, Y. Chen and V.P. Chakka, STRIPES: An efficient index for predicted trajectories, in: *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, ACM, 2004, pp. 635–646. doi:10.1145/1007568.1007639.

[157] J.M. Patel and D.J. DeWitt, Clone join and shadow join: Two parallel spatial join algorithms, in: *Proceedings of the 8th ACM International Symposium on Advances in Geographic Information Systems*, ACM, 2000, pp. 54–61.

[158] K. Patroumpas and T. Sellis, Monitoring orientation of moving objects around focal points, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2009, pp. 228–246. doi:10.1007/978-3-642-02982-0_16.

[159] K. Patroumpas and T.K. Sellis, Managing trajectories of moving objects as data streams, in: *STDBM*, 2004, pp. 41–48.

[160] M. Pelanis, S. Šaltenis and C.S. Jensen, Indexing the past, present, and anticipated future positions of moving objects, *ACM Transactions on Database Systems (TODS)* **31**(1) (2006), 255–298. doi:10.1145/1132863.1132870.

[161] D. Pfoser, Indexing the trajectories of moving objects, *IEEE Data Eng. Bull.* **25**(2) (2002), 3–9.

[162] D. Pfoser and C.S. Jensen, Indexing of network constrained moving objects, in: *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, ACM, 2003, pp. 25–32.

[163] D. Pfoser, C.S. Jensen and Y. Theodoridis, Novel approaches in query processing for moving objects, 2000.

[164] D. Pfoser and Y. Theodoridis, Generating semantics-based trajectories of moving objects, *Computers, Environment and Urban Systems* **27**(3) (2003), 243–263. doi:10.1016/S0198-9715(02)00023-6.

[165] K. Porkaew, I. Lazaridis and S. Mehrotra, Querying mobile objects in spatio-temporal databases, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 59–78. doi:10.1007/3-540-47724-1_4.

[166] F. Porto, J.N. Rittmeyer, E. Ogasawara, A. Krone-Martins, P. Valduriez and D. Shasha, Point pattern search in big data, in: *Proceedings of the 30th International Conference on Scientific and Statistical Database Management*, ACM, 2018, p. 21.

[167] S. Prabhakar, Y. Xia, D.V. Kalashnikov, W.G. Aref and S.E. Hambrusch, Query indexing and velocity constrained indexing: Scalable techniques for continuous queries on moving objects, *IEEE Transactions on Computers* **51**(10) (2002), 1124–1140. doi:10.1109/TC.2002.1039840.

[168] S. Puri, Efficient parallel and distributed algorithms for GIS polygon overlay processing, 2015.

[169] K. Raptopoulou, A.N. Papadopoulos and Y. Manolopoulos, Fast nearest-neighbor query processing in moving-object databases, *GeoInformatica* **7**(2) (2003), 113–137. doi:10.1023/A:1023403908170.

[170] S. Ray, B. Simion, A.D. Brown and R. Johnson, Skew-resistant parallel in-memory spatial join, in: *Proceedings of*

*the 26th International Conference on Scientific and Statistical Database Management*, ACM, 2014, p. 6.

[171] L. Relly, A. Kuckelberg and H.-J. Schek, A framework ofa generic index for spatio-temporal data in C oncert, in: *Spatio-Temporal Database Management*, Springer, 1999, pp. 135–151. doi:10.1007/3-540-48344-6_8.

[172] J.B. Rocha-Junior and K. Nørvåg, Top-k spatial keyword queries on road networks, in: *Proceedings of the 15th International Conference on Extending Database Technology*, ACM, 2012, pp. 168–179.

[173] E. Rslan, H. Abdelhameed and E. Ezzat, An efficient hybridized index technique for moving object database, *Spatial Information Research*, 1–11.

[174] C. Salgado, M.A. Cheema and M.E. Ali, Continuous monitoring of range spatial keyword query over moving objects, *World Wide Web* **21**(3) (2018), 687–712. doi:10.1007/s11280-017-0488-3.

[175] S. Šaltenis, Indexing the positions of continuously moving objects, in: *Encyclopedia of GIS*, Springer, 2008, pp. 538–543. doi:10.1007/978-0-387-35973-1_618.

[176] S. Saltenis and C.S. Jensen, Indexing of moving objects for location-based services, in: *Proceedings 18th International Conference on Data Engineering*, IEEE, 2002, pp. 463–472. doi:10.1109/ICDE.2002.994759.

[177] S. Šaltenis, C.S. Jensen, S.T. Leutenegger and M.A. Lopez, *Indexing the Positions of Continuously Moving Objects*, SIGMOD'00: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, ACM Press, New York, NY, USA, 2000.

[178] H. Samet, J. Sankaranarayanan and M. Auerbach, Indexing methods for moving object databases: Games and other applications, in: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ACM, 2013, pp. 169–180. doi:10.1145/2463676.2465332.

[179] J.C. Shafer and R. Agrawal, Parallel algorithms for high-dimensional proximity joins, in: *Proceedings of the International Conference on Very Large Data Bases*, Institute of Electrical & Electronics Engineers (IEEE), 1997, pp. 176–185.

[180] C. Shahabi, M.R. Kolahdouzan and M. Sharifzadeh, A road network embedding technique for k-nearest neighbor search in moving object databases, *GeoInformatica* **7**(3) (2003), 255–273. doi:10.1023/A:1025153016110.

[181] S. Shekhar, S. Chawla, S. Ravada, A. Fetterer, X. Liu and C.-T. Lu, Spatial databases-accomplishments and research needs, *IEEE transactions on knowledge and data engineering* **11**(1) (1999), 45–55. doi:10.1109/69.755614.

[182] B. Shen, Y. Zhao, G. Li, W. Zheng, Y. Qin, B. Yuan and Y. Rao, V-tree: Efficient knn search on moving objects with road-network constraints, in: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, IEEE, 2017, pp. 609–620. doi:10.1109/ICDE.2017.115.

[183] J. Sheng Min, Efficient geographic information systems: Data structures, Boolean operations and concurrency control, 1990.

[184] S. Shohdy, Y. Su and G. Agrawal, Load balancing and accelerating parallel spatial join operations using bitmap indexing, in: *2015 IEEE 22nd International Conference on High Performance Computing (HiPC)*, IEEE, 2015, pp. 396–405. doi:10.1109/HiPC.2015.43.

[185] D. Šidlauskas, K.A. Ross, C.S. Jensen and S. Šaltenis, Thread-level parallel indexing of update intensive moving-object workloads, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2011, pp. 186–204. doi:10.1007/978-3-642-22922-0_12.

[186] D. Šidlauskas, S. Šaltenis, C.W. Christiansen, J.M. Johansen and D. Šaulys, Trees or grids?: Indexing moving objects in main memory, in: *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2009, pp. 236–245.

[187] D. Šidlauskas, S. Šaltenis and C.S. Jensen, Parallel main-memory indexing for moving-object query and update workloads, in: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, ACM, 2012, pp. 37–48. doi:10.1145/2213836.2213842.

[188] Y.N. Silva, X. Xiong and W.G. Aref, The RUM-tree: Supporting frequent updates in R-trees using memos, *The VLDB Journal – The International Journal on Very Large Data Bases* **18**(3) (2009), 719–738.

[189] B. Singhvi and N. Sarda, Distributed Query Execution Framework for Big Spatial Data, PhD thesis, Indian Institute of Technology, Bombay Mumbai.

[190] A.P. Sistla, O. Wolfson, S. Chamberlain and S. Dao, Modeling and querying moving objects, in: *Data Engineering, 1997. Proceedings. 13th International Conference on*, IEEE, 1997, pp. 422–432.

[191] A.P. Sistla, O. Wolfson, S. Chamberlain and S. Dao, Querying the uncertain position of moving objects, in: *Temporal Databases: Research and Practice*, Springer, 1998, pp. 310–337. doi:10.1007/BFb0053708.

[192] J. Sola, Towards visual localization, mapping and moving objects tracking by a mobile robot: A geometric and probabilistic approach, PhD thesis, 2007.

[193] R. Song, W. Sun, B. Zheng and Y. Zheng, PRESS: A novel framework of trajectory compression in road networks, *Proceedings of the VLDB Endowment* **7**(9) (2014), 661–672. doi:10.14778/2732939.2732940.

[194] S.I. Song, Y.H. Kim and J.S. Yoo, An enhanced concurrency control scheme for multi-dimensional index structures, in: *Proceedings Seventh International Conference on Database Systems for Advanced Applications. DASFAA 2001*, IEEE, 2001, pp. 200–207. doi:10.1109/DASFAA.2001.916379.

[195] S.I. Song, Y.H. Kim and J.S. Yoo, An enhanced concurrency control scheme for multidimensional index structures, *IEEE Transactions on Knowledge and Data Engineering* **16**(1) (2004), 97–111. doi:10.1109/TKDE.2004.1264825.

[196] Z. Song and N. Roussopoulos, K-nearest neighbor search for moving query point, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 79–96. doi:10.1007/3-540-47724-1_5.

[197] J. Su, H. Xu and O.H. Ibarra, Moving objects: Logical relationships and queries, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 3–19. doi:10.1007/3-540-47724-1_1.

[198] J. Sun, D. Papadias, Y. Tao and B. Liu, Querying about the past, the present, and the future in spatio-temporal databases, in: *Proceedings. 20th International Conference on Data Engineering*, IEEE, 2004, pp. 202–213. doi:10.1109/ICDE.2004.1319997.

[199] K. Tabassum, M. Hijab and A. Damodaram, Location dependent query processing – issues, challenges and applications, in: *2010 Second International Conference on Computer and Network Technology*, IEEE, 2010, pp. 239–243.

[200] K.-L. Tan, B.C. Ooi and D.J. Abel, Exploiting spatial indexes for semijoin-based join processing in distributed spatial databases, *IEEE Transactions on Knowledge and Data Engineering* **12**(6) (2000), 920–937. doi:10.1109/69.895802.

[201] L.-A. Tang, Y. Zheng, X. Xie, J. Yuan, X. Yu and J. Han, Retrieving k-nearest neighboring trajectories by a set of point locations, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2011, pp. 223–241. doi:10.1007/978-3-642-22922-0_14.

[202] M. Tang, Y. Yu, Q.M. Malluhi, M. Ouzzani and W.G. Aref, Locationspark: A distributed in-memory data management system for big spatial data, *Proceedings of the VLDB Endowment* **9**(13) (2016), 1565–1568. doi:10.14778/3007263.3007310.

[203] W. Tang and W. Feng, Parallel map projection of vector-based big spatial data: Coupling cloud computing with graphics processing units, *Computers, Environment and Urban Systems* **61** (2017), 187–197. doi:10.1016/j.compenvurbsys.2014.01.001.

[204] E. Tanin, A. Harwood and H. Samet, Using a distributed quadtree index in peer-to-peer networks, *The VLDB Journal – The International Journal on Very Large Data Bases* **16**(2) (2007), 165–178.

[205] Y. Tao, C. Faloutsos, D. Papadias and B. Liu, Prediction and indexing of moving objects with unknown motion patterns, in: *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, ACM, 2004, pp. 611–622. doi:10.1145/1007568.1007637.

[206] Y. Tao and D. Papadias, MV3R-tree: A spatio-temporal access method for timestamp and interval queries, in: *VLDB*, Vol. 1, 2001, pp. 431–440.

[207] Y. Tao and D. Papadias, Efficient historical R-trees, in: *Proceedings Thirteenth International Conference on Scientific and Statistical Database Management. SSDBM 2001*, IEEE, 2001, pp. 223–232.

[208] Y. Tao, D. Papadias and J. Sun, The TPR*-tree: An optimized spatio-temporal access method for predictive queries, in: *Proceedings of the 29th International Conference on Very Large Data Bases-Volume 29*, VLDB Endowment, 2003, pp. 790–801.

[209] Y. Theodoridis, M. Vazirgiannis and T. Sellis, Spatio-temporal indexing for large multimedia applications, in: *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, IEEE, 1996, pp. 441–448. doi:10.1109/MMCS.1996.535011.

[210] G. Trajcevski, Probabilistic range queries in moving objects databases with uncertainty, in: *Proceedings of the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access*, ACM, 2003, pp. 39–45.

[211] G. Trajcevski, O. Wolfson, F. Zhang and S. Chamberlain, The geometry of uncertainty in moving objects databases, in: *International Conference on Extending Database Technology*, Springer, 2002, pp. 233–250.

[212] K. Tzoumas, M.L. Yiu and C.S. Jensen, Workload-aware indexing of continuously moving objects, *Proceedings of the VLDB Endowment* **2**(1) (2009), 1186–1197. doi:10.14778/1687627.1687761.

[213] M. Vazirgiannis and O. Wolfson, A spatiotemporal model and language for moving objects on road networks, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 20–35. doi:10.1007/3-540-47724-1_2.

[214] H. Vo, A. Aji and F. Wang, SATO: A spatial data partitioning framework for scalable query processing, in: *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2014, pp. 545–548.

[215] B. Wang, H. Horinokuchi, K. Kaneko and A. Makinouchi, Parallel R-tree search algorithm on DSVM, in: *Proceedings. 6th International Conference on Advanced Systems for Advanced Applications*, IEEE, 1999, pp. 237–244. doi:10.1109/DASFAA.1999.765757.

[216] H. Wang and R. Zimmermann, Processing of continuous location-based range queries on moving objects in road networks, *IEEE Transactions on Knowledge and Data Engineering* **23**(7) (2011), 1065–1078. doi:10.1109/TKDE.2010.171.

[217] H. Wang, R. Zimmermann and W.-S. Ku, Distributed continuous range query processing on moving objects, in: *International Conference on Database and Expert Systems Applications*, Springer, 2006, pp. 655–665. doi:10.1007/11827405_64.

[218] L. Wang, Y. Zheng, X. Xie and W.-Y. Ma, A flexible spatio-temporal indexing scheme for large-scale GPS track retrieval, in: *The Ninth International Conference on Mobile Data Management (mdm 2008)*, IEEE, 2008, pp. 1–8.

[219] W. Wang, Z. Lv, X. Li, W. Xu, B. Zhang, Y. Zhu and Y. Yan, Spatial query based virtual reality GIS analysis platform, *Neurocomputing* **274** (2018), 88–98. doi:10.1016/j.neucom.2016.06.099.

[220] 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*, IEEE, 2015, pp. 1107–1118. doi:10.1109/ICDE.2015.7113360.

[221] X. Wang, X. Zhou and S. Lu, Spatiotemporal data modelling and management: A survey, in: *Proceedings 36th International Conference on Technology of Object-Oriented Languages and Systems. TOOLS-Asia 2000*, IEEE, 2000, pp. 202–211. doi:10.1109/TOOLS.2000.885919.

[222] 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*, IEEE, 2011, pp. 541–552. doi:10.1109/ICDE.2011.5767861.

[223] F. Wu, K. Fu, Y. Wang, Z. Xiao and X. Fu, A spatial-temporal-semantic neural network algorithm for location prediction on moving objects, *Algorithms* **10**(2) (2017), 37. doi:10.3390/a10020037.

[224] K.-L. Wu, S.-K. Chen and P. Yu, Processing continual range queries over moving objects using VCR-based query indexes, in: *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services. MOBIQUITOUS 2004*, IEEE, 2004, pp. 226–235.

[225] W. Wu, W. Guo and K.-L. Tan, Distributed processing of moving k-nearest-neighbor query on moving objects, in: *2007 IEEE 23rd International Conference on Data Engineering*, IEEE, 2007, pp. 1116–1125. doi:10.1109/ICDE.2007.368970.

[226] Y. Xia and S. Prabhakar, Q+ Rtree: Efficient Indexing for Moving Object Databases, in: *Database Systems for Advanced Applications, International Conference on* IEEE, 2003, p. 175.

[227] Y. Xia and S. Prabhakar, Q+ Rtree: Efficient indexing for moving object databases, in: *Eighth International Conference on Database Systems for Advanced Applications (DASFAA 2003). Proceedings*, IEEE, 2003, pp. 175–182.

[228] J. Xiao, Coarse-grained parallel algorithms for spatial data partition and join processing, in: *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, Springer, 2004, pp. 461–468.

[229] Y. Xing, Load distribution for distributed stream processing, in: *International Conference on Extending Database Technology*, Springer, 2004, pp. 112–120.

[230] X. Xiong, M.F. Mokbel and W.G. Aref, Sea-cnn: Scalable processing of continuous k-nearest neighbor queries in spatio-temporal databases, in: *21st International Conference on Data Engineering (ICDE'05)*, IEEE, 2005, pp. 643–654. doi:10.1109/ICDE.2005.128.

[231] X. Xiong, M.F. Mokbel and W.G. Aref, LUGrid: Update-tolerant grid-based indexing for moving objects, in: *7th International Conference on Mobile Data Management (MDM'06)*, IEEE, 2006, pp. 13–13. doi:10.1109/MDM.2006.102.

[232] T. Xu, X. Zhang, C. Claramunt and X. Li, TripCube: A trip-oriented vehicle trajectory data indexing structure, *Computers, Environment and Urban Systems* **67** (2018), 21–28. doi:10.1016/j.compenvurbsys.2017.08.005.

[233] Y. Yanagisawa, J.-I. Akahani and T. Satoh, Shape-based similarity query for trajectory of mobile objects, in: *International Conference on Mobile Data Management*, Springer, 2003, pp. 63–77. doi:10.1007/3-540-36389-0_5.

[234] X. Yao and G. Li, Big spatial vector data management: A review, *Big Earth Data* **2**(1) (2018), 108–129. doi:10.1080/20964471.2018.1432115.

[235] M.L. Yiu, Y. Tao and N. Mamoulis, The Bdual-Tree: Indexing moving objects by space filling curves in the dual space, *The VLDB Journal – The International Journal on Very Large Data Bases* **17**(3) (2008), 379–400. doi:10.1007/s00778-006-0013-2.

[236] S. You, *Large-Scale Spatial Data Management on Modern Parallel and Distributed Platforms*, 2016.

[237] X. Yu, K.Q. Pu and N. Koudas, Monitoring k-nearest neighbor queries over moving objects, in: *21st International Conference on Data Engineering (ICDE'05)*, IEEE, 2005, pp. 631–642.

[238] Z. Yu, Y. Liu, X. Yu and K.Q. Pu, Scalable distributed processing of K nearest neighbor queries over moving objects, *IEEE Transactions on Knowledge and Data Engineering* **27**(5) (2015), 1383–1396. doi:10.1109/TKDE.2014.2364046.

[239] C. Zhang, F. Li and J. Jestes, Efficient parallel kNN joins for large data in MapReduce, in: *Proceedings of the 15th International Conference on Extending Database Technology*, ACM, 2012, pp. 38–49.

[240] D. Zhang, C.Y. Chow, Q. Li, X. Zhang and Y. Xu, Using parallel spatial mashup model to process k-nn queries, in: *ER Workshops*, 2013.

[241] J. Zhang and S. You, CudaGIS: Report on the design and realization of a massive data parallel GIS on GPUs, in: *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on GeoStreaming*, ACM, 2012, pp. 101–108. doi:10.1145/2442968.2442981.

[242] J. Zhang and S. You, High-performance quadtree constructions on large-scale geospatial rasters using GPGPU parallel primitives, *International Journal of Geographical Information Science* **27**(11) (2013), 2207–2226. doi:10.1080/13658816.2013.828840.

[243] J. Zhang, S. You and L. Gruenwald, Indexing large-scale raster geospatial data using massively parallel GPGPU computing, in: *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2010, pp. 450–453.

[244] M. Zhang, S. Chen, C.S. Jensen, B.C. Ooi and Z. Zhang, Effectively indexing uncertain moving objects for predictive queries, *Proceedings of the VLDB Endowment* **2**(1) (2009), 1198–1209. doi:10.14778/1687627.1687762.

[245] R. Zhang, H. Jagadish, B.T. Dai and K. Ramamohanarao, Optimized algorithms for predictive range and knn queries on moving objects, *Information Systems* **35**(8) (2010), 911–932. doi:10.1016/j.is.2010.05.004.

[246] R. Zhang, D. Lin, K. Ramamohanarao and E. Bertino, Continuous intersection joins over moving objects, in: *2008 IEEE 24th International Conference on Data Engineering*, IEEE, 2008, pp. 863–872. doi:10.1109/ICDE.2008.4497495.

[247] R. Zhang, J. Qi, D. Lin, W. Wang and R.C.-W. Wong, A highly optimized algorithm for continuous intersection join queries over moving objects, *The VLDB Journal – The International Journal on Very Large Data Bases* **21**(4) (2012), 561–586. doi:10.1007/s00778-011-0259-1.

[248] S. Zhang, J. Han, Z. Liu, K. Wang and Z. Xu, Sjmr: Parallelizing spatial join with mapreduce on clusters, in: *2009 IEEE International Conference on Cluster Computing and Workshops*, IEEE, 2009, pp. 1–8.

[249] X. Zhao, J. Shi, M. Du, F. Ni, S. Jiang and Y. Wang, A road-aware spatial mapping for moving objects, in: *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, IEEE, 2018, pp. 1–8.

[250] B. Zheng and D.L. Lee, Semantic caching in location-dependent query processing, in: *International Symposium on Spatial and Temporal Databases*, Springer, 2001, pp. 97–113. doi:10.1007/3-540-47724-1_6.

[251] B. Zheng, W.-C. Lee, K.C. Lee, D.L. Lee and M. Shao, A distributed spatial index for error-prone wireless data broadcast, *The VLDB Journal* **18**(4) (2009), 959–986. doi:10.1007/s00778-009-0137-2.

[252] B. Zheng, K. Zheng, X. Xiao, H. Su, H. Yin, X. Zhou and G. Li, Keyword-aware continuous knn query on road networks, in: *2016 IEEE 32Nd International Conference on Data Engineering (ICDE)*, IEEE, 2016, pp. 871–882. doi:10.1109/ICDE.2016.7498297.

[253] K. Zheng, S. Shang, N.J. Yuan and Y. Yang, Towards efficient search for activity trajectories, in: *2013 IEEE 29Th International Conference on Data Engineering (ICDE)*, IEEE, 2013, pp. 230–241. doi:10.1109/ICDE.2013.6544828.

[254] K. Zheng, H. Su, B. Zheng, S. Shang, J. Xu, J. Liu and X. Zhou, Interactive top-k spatial keyword queries, in: *2015 IEEE 31st International Conference on Data Engineering*, IEEE, 2015, pp. 423–434. doi:10.1109/ICDE.2015.7113303.

[255] K. Zheng, Y. Zheng, X. Xie and X. Zhou, Reducing uncertainty of low-sampling-rate trajectories, in: *2012 IEEE 28th*

*International Conference on Data Engineering*, IEEE, 2012, pp. 1144–1155. doi:10.1109/ICDE.2012.42.

[256] Y. Zhong, J. Han, T. Zhang, Z. Li, J. Fang and G. Chen, Towards parallel spatial query processing for big spatial data, in: *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, IEEE, 2012, pp. 2085–2094. doi:10.1109/IPDPSW.2012.245.

[257] C. Zhou, Z. Chen and M. Li, A parallel method to accelerate spatial operations involving polygon intersections, *International Journal of Geographical Information Science* **32**(12) (2018), 2402–2426. doi:10.1080/13658816.2018.1508689.

[258] K. Zhou, G. Tan and W. Zhou, Quadboost: A scalable concurrent quadtree, *IEEE Transactions on Parallel and Distributed Systems* **29**(3) (2018), 673–686. doi:10.1109/TPDS.2017.2762298.

[259] X. Zhou, D.J. Abel and D. Truffet, Data partitioning for parallel spatial join processing, *Geoinformatica* **2**(2) (1998), 175–204. doi:10.1023/A:1009755931056.

[260] H. Zhu, X. Yang, B. Wang and W.-C. Lee, Range-based nearest neighbor queries with complex-shaped obstacles, *IEEE Transactions on Knowledge and Data Engineering* **30**(5) (2018), 963–977. doi:10.1109/TKDE.2017.2779487.

[261] X. Zhu, A study on data consistency in spatial database system, *International Archives of Photogrammetry and Remote Sensing* **33**(B4/3; Part 4) (2000), 1252–1256.

[262] R. Zimmermann, W.-S. Ku and W.-C. Chu, Efficient query routing in distributed spatial databases, in: *Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems*, ACM, 2004, pp. 176–183.