Accepted Manuscript

Multi-dimensional data indexing and range query processing via voronoi diagram for internet of things

Shaohua Wan, Yu Zhao, Tian Wang, Zonghua Gu, Qammer H. Abbasi, Kim-Kwang Raymond Choo

PII: S0167-739X(18)31195-6
DOI: https://doi.org/10.1016/j.future.2018.08.007
Reference: FUTURE 4391

To appear in: *Future Generation Computer Systems*

Received date : 15 May 2018
Revised date : 24 July 2018
Accepted date : 5 August 2018

Please cite this article as: S. Wan, et al., Multi-dimensional data indexing and range query processing via voronoi diagram for internet of things, *Future Generation Computer Systems* (2018), https://doi.org/10.1016/j.future.2018.08.007

Multi-dimensional Data Indexing and Range Query Processing via Voronoi Diagram for Internet of Things

Shaohua Wan[a], Yu Zhao[b], Tian Wang[c], Zonghua Gu[d,*], Qammer H. Abbasi[e], Kim-Kwang Raymond Choo[f]

[a]*School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan, 430073 China*
[b]*Department of Computer Science, Technische Universität München, Munich, 80333 Germany*
[c]*College of Computer Science, Huaqiao University, Xiamen, 361021 China*
[d]*College of Computer Science, Zhejiang University, Hangzhou 310027 China*
[e]*School of Engineering, University of Glasgow. G128QQ Glasgow, U.K.*
[f]*Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249-0631, USA*

Abstract

In a typical Internet of Things (IoT) deployment such as smart cities and Industry 4.0, the amount of sensory data collected from physical world is significant and wide-ranging. Processing large amount of real-time data from the diverse IoT devices is challenging. For example, in IoT environment, wireless sensor networks (WSN) are typically used for the monitoring and collecting of data in some geographic area. Spatial range queries with location constraints to facilitate data indexing are traditionally employed in such applications, which allows the querying and managing the data based on SQL structure. One particular challenge is to minimize communication cost and storage requirements in multi-dimensional data indexing approaches. In this paper, we present an energy- and time-efficient multidimensional data indexing scheme, which is designed to answer range query. Specifically, we propose data indexing methods which utilize hierarchical indexing structures, using binary space partitioning (BSP), such as kd-tree, quad tree, k-means clustering, and Voronoi-based methods to provide

*Corresponding author
Email addresses: shaohua.wan@ieee.org (Shaohua Wan), yu.zhao@tum.de (Yu Zhao), wangtian@hqu.edu.cn (Tian Wang), zgu@zju.edu.cn (Zonghua Gu), Qammer.Abbasi@glasgow.ac.uk (Qammer H. Abbasi), raymond.choo@fulbrightmail.org (Kim-Kwang Raymond Choo)

more efficient routing with less latency. Simulation results demonstrate that the Voronoi Diagram-based algorithm minimizes the average energy consumption and query response time.

Keywords: Range query processing, multi-dimensional data indexing, Voronoi diagram, IoT energy efficiency

---

1. Introduction

Internet of Things (IoT) has many applications in our society, which is not surprising given the capability to facilitate the collection and analysis of a broad range of information in our physical environment (e.g. smart cities, smart vehicles, and smart factories). For example, multi-attribute sensors collaboratively and periodically collect data from their respective environment, and such data are generally multi-dimensional. However, the diversity and ever-increasing volume of data from IoT applications compound the challenge in processing and making sense of such multi-dimensional data. For example, how do we design an energy-efficient spatial index structure to search the multi-attribute sensors in our constantly evolving technological landscape? Range query is a viable solution, which has been used in a number of topics, such as area locations, sizes and aggregated data of areas (min, max, average,...), particularly in mobile applications.

Range queries represent a typical database operation by which one can retrieve stored data that satisfies a specific set of interval-based constraints, such as temperature (e.g. between t_1 and t_2), humidity (e.g. between h_1 and h_2) and light condition (e.g. between l_1 and l_2). These constraints may refer specifically to data-values of some particular tuples of interest, or in the context of spatial-query processing, the locality-bounds of the data.

Spatial-query processing is particularly relevant in a large wireless sensor network (WSN) environment, as the region of interest may not span the entire WSN geographic coverage. As an example, a typical range query can be stated as follows: "retrieve the locations of the nodes, where the temperature is between 90F and 110F". More formally, a range query bears the following type of formulation: "retrieve all the records for which a subset of their attribute values satisfy a set of interval-based constraints c". When the range query has a small life-span or is about simple instantaneous events, constructing routing structures in existing approaches is achievable [1]. However, in many real-world scenarios, the queries are continuous in nature, (i.e. monitoring of some phenomena over a long period). These types of queries are generally referred to as range-monitoring queries, where the answer can change over time and such changes (and not the actual values) need to be reported to the query initiator.

There are, however, a number of challenges in designing a range monitoring query mechanism for a resource constrained WSN. For example, continuous sampling of the environment for prolonged periods of time in an attempt to capture the changes in state can be extremely energy consuming. In addition, when the environment being monitored is highly dynamic, the transmission of an excessive number of updates, either directly or through intermediary aggregations nodes, has several adverse effects, such as increased delay/latency of the response and increased energy consumption. Clearly, inefficient range query approaches can affect the network lifetime (NL) of the underpinning WSN environment, where NL is defined as the maximum total time period from the initial deployment until the network connectivity or coverage is lost. Real-time query/message routing in WSN considering power/energy consumption and NL issues is an active research topic [2, 3].

We have presented prediction techniques and aggregation trees with or without synopsis in our previous work [4, 5, 6]. However, most of existing approaches focus on only one or two particular characteristics, such as how fast the phenomena changes over time and spatial-variability, as well as assuming that these characteristics do not change over time. In practice, one may need an additional flexibility, in the sense that a range monitoring query should be able to adapt to changing network or phenomena conditions, by means of workload-balancing, reconfigurable routes [7, 8], etc. This is the focus of our proposal in this paper (see Section 3).

We also observe that the issue of minimizing energy and bandwidth consumption resources by lowering the minimum required coefficient levels has not been formalized and addressed in the literature. Therefore, in this paper, we approach this issue from a scalability perspective and devise solutions for large-sized WSN. In addition, for mobile object identification and tracking, we will investigate the extent in which the size of the moving targets can influence the results in a practical setting. Firstly, to obtain the dimensionality information of the objects that are detected is a problem on its own. Thus, we will employ a mix of existing techniques, such as triangulation and dead-reckoning. We believe that estimating the size of the targets can lead to more effective solutions for the tracking, counting and identification problem of moving objects. Secondly, we will develop efficient distributed data indexing algorithms for the widely used spatial-temporal range monitoring queries, considering the context of each syntactic variation. Each syntactic construct will be incorporated as extensions of the TinySQL, and the corresponding processing algorithms will be integrated with the query processing engine of the TinyDB (see Figure 1). Also, we will adapt our centralized approach for the processing of dynamical topological predicates in WSN settings, by providing an alternative, scalable, distributed implementation.

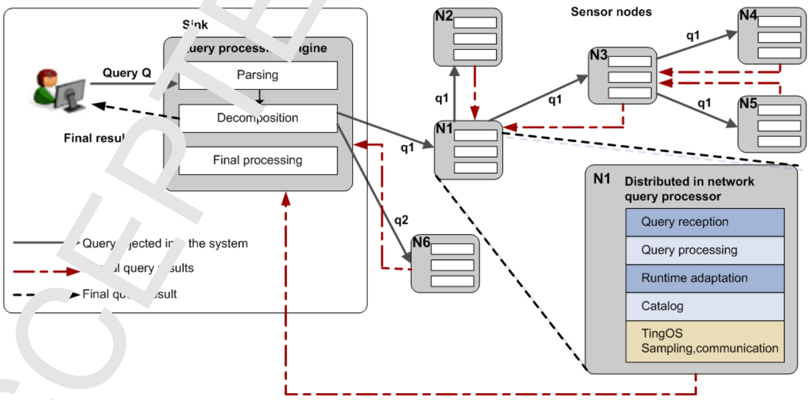


Figure 1: Query processing mechanism with the introduction of TinyOS.

Specifically in this paper, in order to efficiently optimize the use of the network resources and improve the performance of energy consumption and query response time in WSN, we propose a novel range data aggregation approach by exploiting spatial structures of sensory data. The contributions of this paper are summarized as follows:

- We propose effective multidimensional data indexing structures to help process spatial queries efficiently. This results in a high dimensional data indexing architecture for addressing existing problems and enables us to present approaches which are more suited in mobility and spatial continuous range queries, than those proposed in previous works. In this scheme, the indexing scheme equally handles both types of information, and aggregates them in an energy efficient manner. Our approach also includes a hierarchical in-network storage that is capable of responding to different queries in a timely fashion, with immediate answers to approximate queries and some types of exact queries.
- In order to determine whether the proposed data indexing algorithms are sufficiently generic for commonly used spatial query processing, we evaluate on four data structures, namely: kd-tree, quad-tree, k-means clustering and Voronoi diagram (VD). VD data indexing model is suitable for general queries operations, which can, for example, be applied to process location-based service in the cells in O(log n) time.

In the next two sections, we present related literature, and relevant materials on spatial query and key factors that may affect query processing. Section 4 presents our proposed architecture for spatial query processing. In the section, we also evaluate the applicability of the indexing algorithm on four data structures. Section 5 presents the findings from our experimental simulation analysis and its' performance analysis. Finally, we conclude the paper in Section 6.

2. Related work

The quality of a query answer, which we represent by its confidence level, can be improved in a naive manner by committing more resources towards query processing (e.g., increasing the number of nodes involved in query answer and the frequency these nodes participate). In other words, we can increase the confidence level of the answer of a query if we are willing to devote more energy and bandwidth resources. However, focusing on the quality of an answer for a particular query should also take into consideration the Quality of Service (QoS) provided by the underpinning network. QoS can be expressed using the average, median and standard deviation of the confidence levels of the answers of all possible queries and the lifetime of the sensor networks. Clearly, it is desirable to have a sensor network that is able to provide "adequate" results for a prolonged period of time, rather than minimum-error results for a very short time. In other words, we should be able to accept a slightly lower confidence level in order to benefit from a longer sensor infrastructure's shelf life.

In the literature, there are a number of definitions for the lifetime of a WSN, such as the time the first node in the network dies, the time when a preset percentage of the nodes die, and the time the network loses connectivity [9]. These definitions are, in fact, instances of a general criteria by which the lifetime of a network is considered expired (i.e. QoS degradation of a WSN below some acceptable threshold). The degradation in QoS can also be expressed either in terms of lowered network resolution or by not being able to route query answer to query initiator, in a timely manner, due to dis-connectivity or routing holes issues. Either way, various choices of the admissible QoS thresholds can be mapped to one of the former definitions of lifetime. Unfortunately, QoS thresholds are application specific and their relevance can only be discussed in the context of their application. Arguably, a slightly more generic definition of the lifetime, which is not explicitly bound to the specifics of the covered phenomenon is the following: the time interval during which the confidence levels of the query answers that the network can provide are above some predefined thresholds. Our work will rely on the confidence-level criteria, since it provides a clearer connection between the query answer's accuracy and the lifetime.

These ideas are not necessarily new as they have been expressed differently in various contexts, albeit not by means of confidence levels. For example, the authors in [10, 11, 12] proposed optimal transmission scheduling for point-to-point routing with end-to-end delay constraints that relies on delay margins to extend the NL. In a sense, it fits the definition of the lifetime that we propose, in terms of confidence levels, since they are leveraging delay margins for lifetime purposes. This translates into trading (lowering) the confidence levels requirements, within admissible bounds, for the same purpose. A separate class of algorithms concerning the balancing of workload by leveraging end-to-end delay margins [13, 14, 15] is similar to our proposed approach. Other lifetime extension techniques rely on various data reductions (e.g. data aggregation and filtering), in order to reduce the most energy-expensive function of the sensor nodes communication [16, 17, 18]. Some of these techniques are lossy, with controlled error bound, which leverage the data filtering principle. Lifetime extending techniques have been proposed for all networking layers in WSN, namely: application, network, link and physical. These , in essence, perform the same task: trade answering precision (confidence levels) for energy efficiency.

The importance of augmenting query responses with confidence levels has also been studied. For example, authors of [12, 19, 20] explore how confidence levels can affect data management decisions, and their approaches rely on the static and dynamic adjustment of the transmission parameters in order to achieve the highest confidence level when some specific application request. Another related work is the QUASAR project [21], which highlights the need to leverage application's imprecision to minimize resource consumption and to represent and handle the flow of data of varying quality. The authors acknowledged the difficulty of interpreting the results of complex queries by relying solely on absolute error margins, tied to the application environment specifics. However, significant energy and bandwidth resources can be further minimized by lowering the minimum required coefficient levels, which has not been addressed in the existing literature.

Most existing approaches are designed for small-sized, personal wireless sensor network of sensors. In addition, existing lifetime extending algorithms generally rely on the assumption that the level of admissible "imprecision" is known *apriori*, by being hardcoded, pre-configured in the devices or by being explicitly declared in the query statement. The first method is less flexible, but nevertheless it should be adopted at all times and used as the default imprecision margins when users do not specify their own. The second method provides the most flexibility in specifying the tolerance margins, but its performance is limited by the subjective imprecision margins the user tolerates and specifies. Also, it requires the users to have domain expert knowledge about the intrinsic parameters of the phenomena that is being monitored in order to choose these parameters efficiently. This method should be employed only when absolute precision is required. For this, it is much easier for the user to be able to alter (increase or decrease) the default minimum confidence level of the expected answer of a prospective query, which is simpler to understand, normalized value. Under these considerations, we intend to investigate how to prolong the network's lifetime without compromising on trade the accuracy of the answer.

Another important aspect pertaining to the tracking of mobile objects queries is the choice of an adequate mobility model (e.g. periodical, such as location, time, and velocity, updates generated by mobile units [22, 23], and fully-known future trajectories [24, 25, 26]). The main reasons are: (1) limited sensing coverage, memory and power budgets of the nodes in the sensor networks; (2) the objects that are tracked need not be cooperative in the sense of communicating their (location, time) information. Some existing works for spatial-temporal data for mobile objects in WSN may be readily adapted for processing a NN-query. For example, the processing of the following query: Q-NN1:"retrieve Nearest Neighbor of object o1 between 2:30 and 3:00" can be achieved with minor modification of some of the results in [27] by enforcing a detection of the objects within the proximity of the tracked-object (o_1) and properly updating the answer when needed. The local changes of the answer can subsequently be transmitted to the (static and or mobile) sink. However, scalability becomes a problem when processing the K-NN variant or, for that matter, the all-pairs-NN [28]. In general, the approaches proposed in the in the Moving Object Database (MOD) literature [29, 30, 31] cannot be directly "translated" into sensor networks settings.

3. Range Queries

There are a number of known challenges when processing spatial-temporal range queries in WSN settings, such as those illustrated in Figure 1. Let us assume that the following query is posted in a dense network: Q-R_1:"retrieve the number of distinct objects inside the region R between 12:00 and 12:30". One observation is that some objects, like o1, will need to be tracked for the purpose of correct maintenance of the query like Q-R_1 even when they exit the region of interest for the query. Namely, unless o1 is tracked and its identity maintained by the sensors outside R, it may (leave or) re-enter the region more than once during the time-interval of interest [12:00, 12:30] and result in an incorrect update to the answer-set. Another important observation is that, although Q-R_1 seems to be clearly stated, its syntax is, in a sense, not quite complete. Note that one of the features offered by TinySQL is that users can specify certain constructs that influence the processing, such as the sampling frequency and the duration of a given query.

In the case of Q-R_1, although its nature is continuous, distinct syntactic variations will impose different processing vs. communication trade-offs. For example, (1) report the full answer at the end of the time-interval of interest; (2) report the initial answer and present cumulative updates every 5 minutes; or (3) report the initial answer and present updates whenever the answer changes.

There have been attempts [29, 30, 31] to design efficient reactive management of topological predicates. In such solutions, it is necessary to manage the continuous and persistent conditions in order to measure the satisfiability of such estimation in mobile and dynamic environments. In spatial settings, the alongness property has also been investigated both from topological (the 9-intersection model in [32]) and spatial database [33] perspectives. When it comes to the "alongness" in mobile environments, in reality one cannot expect that a mobile object can move exactly along a particular topological curve (e.g. a river). Thus, a distance threshold d has been introduced (i.e. for as long as the object is within distance from a given 2D polyline P, the object will be assumed to be moving along). Also, one needs to check whether a predicate is satisfied within a portion t of a time-interval $[t_1, t_2]$. As a particular example, consider the following request which is important in scenarios like adversarial environment such as battlefields: Q-R_2: "Notify me when the object obj_1 is moving along the polyline P and within distance d less than 90% of the time between 5:00 and 5:30".

Figure 2 shows an example scenario, where each circle indicates some update sent to the MOD server (e.g. location or time update). In this example, we assume that they are sent every two minutes. A blank circle denotes (location, time) pair of no interest for processing Q-R_2 because the value of their time component is outside the time-interval of interest for Q-R_2 ([5:00, 5:30]).

The moving towards predication is concerned with detecting if a particular mobile object is continuously moving towards a given static entity, like a point-object, region or a polyline. To illustrate the aspects of the reactive behavior that are of interest regarding this predicate, let us consider the following query: Q-R_3 "notify me when the object obj_2 is moving towards the landmark LM continuously for 5 minutes between 5:00 and 5:30". As observed, Q-R_3 is satisfied at 5:18 because between 5:12 and 5:18 the object was continuously moving towards LM for 6 minutes.

Current solutions for the evaluation of these topological predicates, however, assume that the location information are sent to a central server before being processed. Such centralized approaches are not suited in a distributed WSN, particularly in dealing with spatial-temporal tracking queries. Specifically, we require an approach that provideds primitives for implementing the moving along and moving towards dynamical topological predicates in WSN. Hence, we implement a dead-reckoning algorithm for the purpose of estimating the future locations of the mobile objects. This is necessary to decide when and which node should transmit location updates to the sink nodes for processing, and push the decision processing logic for these topological predicates towards the nodes that are currently active in the process of tracking a particular moving object, in order to achieve scalability and de-centralization of the original algorithms.

One main task of a WSN is to respond to the triggered spatial queries. The queries may inquire values of the sensed phenomena, either in the entire field or in a specific region. They may also inquire the location from which a value, or a range of values, were reported. Spatial queries are more likely to inquire information about the overall behavior rather than specifics. Also, the reported values of sensor nodes are generally not accurate due to imperfection and other physical aspects. Hence, approximate queries are more suited for WSN, where the query contains a field to specify an acceptable accuracy level. Hence, queries are considered as predicates with attributes, as follows: Q(P, L, R, T), where:

P means the sensory phenomenon (e.g., Temperature, Light)

L means a sensor location

R means the query within the sensed geometric range (R), and/or, either value range within the sensed values or an extreme (M, where M = min or M = max).

T T means the required time for the query response.

An query example with range constraint would be straightforwardly translated to an SQL-like syntax:

SELECT MAX(Sensor.Temerature) FROM Sensor WHERE Sensor.Location INSIDE RECTANGLE [0, 0], [100, 100] AND Sensor.Time BETWEEN 12/21/2017 and 12/22/2017.

4. Proposed Spatial Range Query Processing Approach

Firstly, we intend to investigate the benefits of adopting a modified version of the probabilistic uncertainty model which will support singleton query results but augmented with a simple confidence coefficient, rather than a confidence interval. To support out intentions, let's consider the following example: in a military application, a user submits the following informal query: "retrieve the number of enemy vehicles that have been moving toward base station B1 in the last M minutes and are less than D miles away." The user, which can be a field combatant, knows that if, say, n or more enemy vehicles are moving towards, then he needs to trigger an alarm. Under a point uncertainty model, the answer could be, for example, "n", which may or may not be correct. Adopting the interval uncertainty model, the answer of the query may be, for example, represented as a numerical interval I=$[n_1, n_2]$, $n_1 < n < n_2$, which, considering the particularities of this query, will not provide sufficient information for the combatant to trigger the alarm. The implications of such lack of information can be even deeper: let's imagine that a meta-trigger is placed in the network monitoring the number of enemy vehicles that are moving towards, and the specification of the trigger indicates that an alarm should be triggered when n such vehicles are detected. Only a probabilistic uncertainty model may provide insight onto the likelihood of each possible value in the given answer interval, but, as we have already mentioned, it can be difficult reason in real time and time critical applications, especially when the answer is not as trivial as the one we considered. We argue that an answer on the form "n enemy vehicles" with confidence level l ($0 \leq c \leq l$) represents a better representation on the answer for most applications and we intend to develop a methodology for query processing with confidence coefficients, with a specific focus on spatial-temporal range monitoring queries. As a justifiable argument is that we can configure the meta-triggering mechanism with a singleton threshold lt for the answer is $l \geq lt$ the alarm should be fired. Moreover, this threshold can be unanimously be set as default value for all the mete-triggering mechanism that are dispatched in the network, regardless of the specifics of the queries.

We will analyze spatial queries in stages for a better understanding of them. As is known to all, different numbers of stages can be defined for spatial query processing in WSN. However, as we have previously stated, these stages can be further broken down into simpler ones. In this paper, we would specifically analyze spatial queries from the following six steps: 1) pre-processing; 2) forwarding; 3) dissemination; 4) sensing; 5) aggregation; and 6) return (see Figure 2). In the step of pre-processing, queries are formatted so that they can be diffused via the intermediate nodes. Such procedure is usually done in user's computer, as there are more resources on this computer than sensor nodes. Also, in the stage of pre-processing, it is a necessity and a must to perform application-independent task, for example, representing the information with max appropriateness and suitability, so that the queries can be more efficient and less packets will be taken up. Then comes to the forwarding and dissemination stages, where queries are forwarded and spread to the region of interest(RoI) from the Originator (the first node that the query can be received in the network). It is noteworthy that these queries are only forwarded and propagated to nodes within the RoI. This is different from traditional query processing, which requires the dissemination of queries to all nodes in the WSN through Flooding. Specifically, the purposes to forward and disseminate queries to all nodes within the RoI are to ensure the best energy consumption and minimized the number of packets that are transferred in the WSN. Then moves on to the sensing stage in which the data required by the query are collected by the nodes within the RoI and are then transmitted to the sink node to calculate the query result.

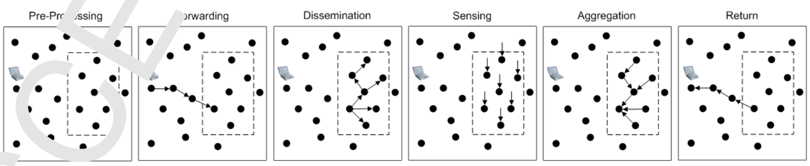


Figure 2: Data aggregation of spatial query processing.

4.1. Kd-tree Query Processing Routing

The distributed index structure drives efficient processing of queries and imposes restrictions on the number of sensor nodes involved. The query problem, in effect, is finding the data within a specified query range or interval. Usually, we will regard numerical fields of objects as coordinates (where a point set is stored in higher dimensions). A set of n points inside a 1D query range can then be answered in a fast manner, provided that they are preprocessed on the real line. That is to say, these points $p_1, ..., p_n$ will be known in advance and the query [x, x_0] is known later. To solve the query problem, a data structure, a query algorithm, and a construction algorithm are often used.

Kd-tree represents d-dimensional trees which are general, simple, and arbitrary dimensional. However, its complexity analysis result may not be very good for asymptotic search. Kd-tree has extended 1D tree by alternate use of xy-coordinates to split and cycled the dimensions in k-dimensions. Specifically, it splits x-coordinate by a vertical line so that half of the points are right and the other half are left; it splits y-coordinate by virtue of a horizontal line so that half of the points are above and the other half are below (see Figure 3). Each node within this binary tree has two values: split dimension and split value. In case it is split along x at the coordinate s, points with x-coordinate $\leq s$ are included in the left children and the others are included in the right children. The same principle applies to the split along y. If O(1) points remain, they will be put in a leaf node, with the data pointing at leaves only and internal nodes for splitting and branching. In order to balance trees, median coordinate is used since splitting-median itself is accessible in either half. The height of the tree is guaranteed to be O (log n) by using median to split. Then comes two options: 1) cycling through the splitting dimensions; 2) making data-dependent choices (such as: selecting dimension with max spread).

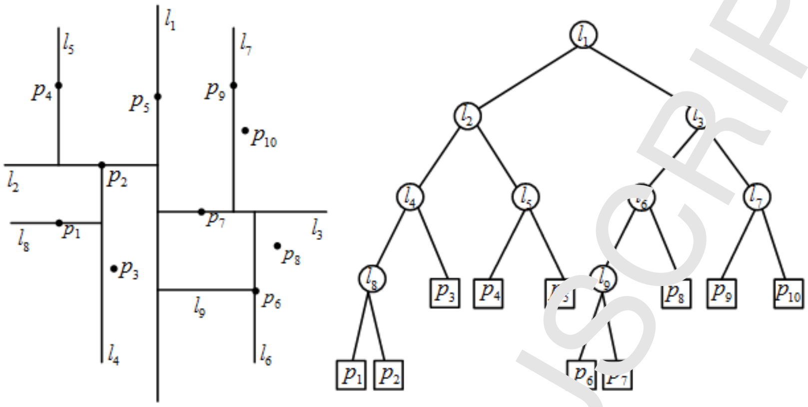


Figure 3: Kd-tree Query Construction.

Kd-tree has a space subdivision by the way that an x- or y-aligned cut is introduced for each node, and the points on two sides of the cut will then be passed to nodes in left and right children. The subdivision is composed by rectangular regions or cells that may be unbounded. Root corresponds to the entire space where each child shares one of the half-spaces. Different from that, leaves correspond to the terminal cells. A general partition BSP is a special case. Its structure can be constructed in O($n \cdot logn$) time in a recursive way. Then, points need to be presorted by x and y-coordinates, and such two sorted lists need to be cross-linked. The way to find the x-median is to scan the x list. Then it comes to the splitting of the list into two, and the use of cross-links for splitting of y-list in O(n) time.

4.2. QUAD-TREE PROCESS ROUTING

In a quad-tree, there are exactly four children inside each internal node. In such a tree data structure, each node represents a bounding box that has some part of indexed space covered, and has the entire area covered by root node. In the structure of a quad-tree, the depth is set as O(log n) for the uniform sensor distribution. It is simple to insert data into a quad-tree, with the following three steps taken: 1) starting at the root and identifying which quadrant your point stays; 2) finding a leaf node through recursing to that node and repeating;

Algorithm 1 Kd-TreeQuery

Require:

```
P,R P denots a kd-tree's root and R denotes a range;
```

Ensure:

```
All the leaves nodes below P which are within the range;
if P is a leaf node then
    Output the nodes stored at P if it is in R;
else if area(lc(P)) is completely located in R then
    OutputSubtree(lc(P));
else if area(lc(P)) crosses R then
    Kd-TreeQuery(lc(P),R);
    if area(rc(P)) is completely located in R then
        OutputSubtree(rc(P));
    else if area(rc(P)) crosses R then
        Kd-TreeQuery(rc(P),R);
```

3) putting your point into the list of points of that node. In case that the list exceeds the max number of some elements that are pre-determined, the node needs to be split and then the points need to be moved into the correct sub-nodes. To query a quad-tree, the following steps are needed: 1) starting at the root and examining each child node; 2) checking if child node intersects with the query area. If it does, what needs to do next is recursing to that child node. Whenever a leaf node is found, each entry needs to be examined to make it clear if it intersects with the area being queried for, then return to it if it does. Then, we can construct the quad-tree in a recursive way, given a list of particle positions.

Figure 4 depicts the structure of a quad-tree, where, obviously, all inter nodes have four children.

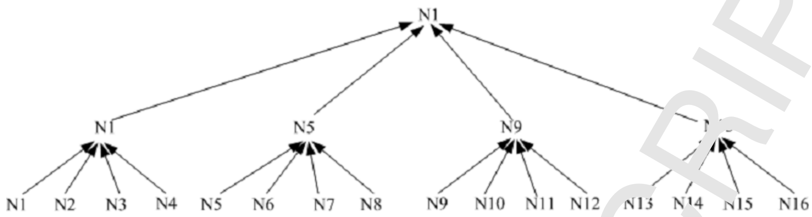


Figure 4: Quad indexing tree.

4.3. K-MEANS CLUSTERING BASED QUERY PROCESSING ALGORITHM

Among the many different choices of learning algorithms, k-means is the most popular one being adopted for clustering. Considering the fact that highly correlated measurements are obtained from sensors that are closely located, we purport to cluster nodes in accordance with the locations of those nodes and the similarity of their physical attributes. In addition, as previously stated, it is unavoidable that a great amount of redundancy exists with regard to the readings from each sensor over time. Together those constitute the foundations for modeling the spatiotemporal correlation in data. Therefore, what we need to do is to define a feature vector for each node so that entire behavior of that node can be well reflected. Employing K-means algorithm is helpful in electing the cluster head in an efficient manner, and in particular, selecting an appropriate cluster head can exert great impact on the reduction of energy consumption and the improvement of NL (see Figure 5). This is because the more demanding the accuracy and computational requirements are, the greater energy consumptions will be. Otherwise, developed systems might be used in replace of K-means algorithm, and then the learning task is performed by centralized and resource capable computational units.

It is found that the widely employed clustering algorithms in WSN are good for the clustering of sensor nodes so as to meet the objectives of scalability and energy efficiency as well as the election of the head of each cluster. In recent years, although an extensive number of clustering routing protocols have been put forward for WSN [34, 35, 36, 37], little of them have considered the use of the data science clustering techniques in a direct way. Instead, those data clustering techniques are used for the purpose of finding the similarities or correlations in data between neighboring nodes, and partition sensor nodes into clusters accordingly. The following is the application of K-means in wireless networks. In [34, 35, 36, 37], the sensory data is clustered via the distributed k-means clustering algorithms, and then is aggregated and transmitted towards a sink node. The purpose of such summary of data is to ensure the reduction of communication transmission and processing time, as well as the reduction of energy cost of the sensor nodes.

It is inappropriate to adopt a centralized method (collecting data from sensors as predetermined and transmitting the collected data to a server for storage and querying) for query processing in WSN. This is because in such conditions, valuable resources will be occupied for transmitting large quantities of raw data to the cloud system, and in most cases, the transfer can be redundant. In fact, it is a must to save energy in sensor networks so that the lifetime of sensors can be extended, as those sensors are usually recharged by batteries with low capacity. Considering that data processing is a lot cheaper than wireless communication cost, it is not a necessity of transmitting all data to sink node for processing. Instead, part of data can be transmitted from the sink to the base station. Under such conditions, the power dissipation can be reduced.

The purpose of K-means is to partition n observations to k clusters, so that observations are respectively grouped to the clusters with the nearest mean, which serve as the prototypes of the clusters. Assume that within a set of values $(x_1, x_2, ..., x_n)$, each one of them is a multi-dimensional real vector. Then a k-means clustering is employed to divide such n values into k ($k \leq n$) sets s=$s_1, s_2, ..., s_k$, thereby minimizing the sum of squares within the cluster.

The following three parts composes the query processing algorithm: 1) K-means clustering algorithm, 2) energy-efficient query transmission and 3) result collection. Upon the user's specific request on precision, head nodes are selected to respond to the user's query, and results are collected in an energy-efficient way through the clustering algorithm. Based on the simulation results, it implies that

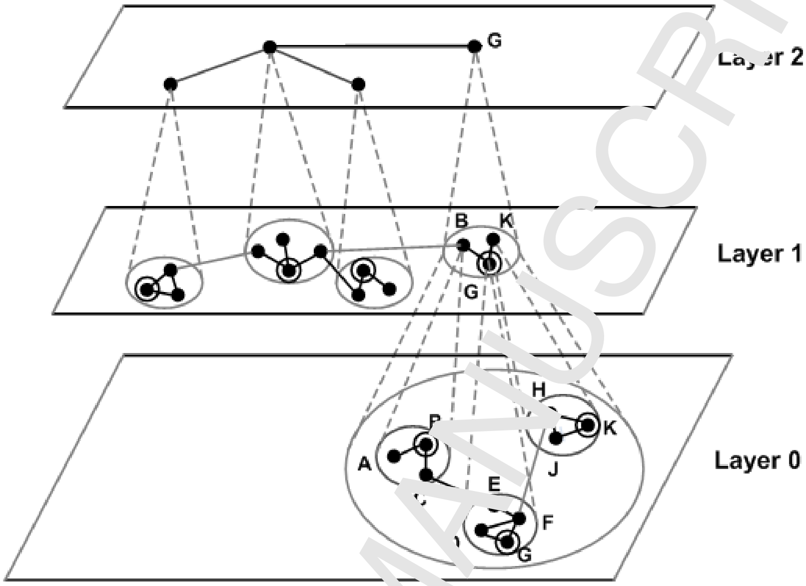


Figure 5: Data aggregation example in a clustered architecture, where the nodes are marked as first level and second level cluster heads.

Algorithm 2 K-means Clustering

Select k cluster heads of the n sensors;
Associate each node to the closest cluster head;
Calculate the initial cost (sum of the Euclidean distances of each point to its cluster head);
repeat
    Swap a cluster head with a non-cluster head point;
    Re-compute the cost (sum of distances of points to their cluster heads);
until the total cost of the configuration increased

ACCEPTED MANUSCRIPT

with head node selection of K-means clustering, the query processing algorithm not only can ensure a more precise result but also can reduce more energy consumption than the other algorithms. Specifically, each node performs the task of sensing and then each node will send the data to its cluster head. Then the cluster head reconstructs the data sent from all nodes, before its averaging all measurements for the reduction of dimensionality. Finally, the cluster head will compress those data by performing it on the average subsequent to which the data will be sent to the sink node.

4.4. VD-based multi-dimensional data indexing Algorithm

In computational geometry, a Voronoi diagram (VD) is one of the most significant models, and widely used to divide a plane into regions which relies on the points in a definite subset of the plane. Assume $P = p_1, p_2, ..., p_n$ to be a set of nodes in the plane, called sites. The VD divides the two-dimensional continuous space (or any dimensional space) into closed subspaces by equidistant partitioning between any two points, which is called Voronoi cell. The Voronoi cell for $p_i, V(p_i)$, is defined to be the set of nodes q in the plane whose Euclidean distances between p_i and q are smaller than that to any other site. That is, the formal representation of the Voronoi cell for p_i is:

$$V(p_i) = \{q | dist(q, p_i) \leq dist(q, p_j), \forall p_j \in P, i \neq j\} \tag{1}$$

Clustering a set of sensors tries to categorize the nodes into their respective clusters according to the distance to cluster head. In monitoring applications of IoT, VD partitioning space into dissimilar regions facilitates the sensing task to the different regions in a distributing way. Sensors from different clusters sense, process, and transmit data to the intra-cluster head respectively, and then inter-clusters efficiently perform data-processing to the higher level. This paper has explored a distributed clustering and hierarchal algorithm which layers sensors in a large volume Voronoi cells based WSN for the purpose of reducing the total energy consumption. The key point of this algorithm is VD's construction, a k-clustering of P problem, which is to find k clusters (subsets) by partitioning

P, $C_1, C_2, ..., C_k$ (see Figure 6). Let us assume $\mu(C)$ denotes an intra-cluster criterion, and $\delta(C_1, C_2, ..., C_k)$ means the inter-cluster criterion. Theoretically,

$$\delta(C_i, C_j) = max\{dist(p, q)|p \in C_i, q \in C_j, i \neq j\} \tag{2}$$

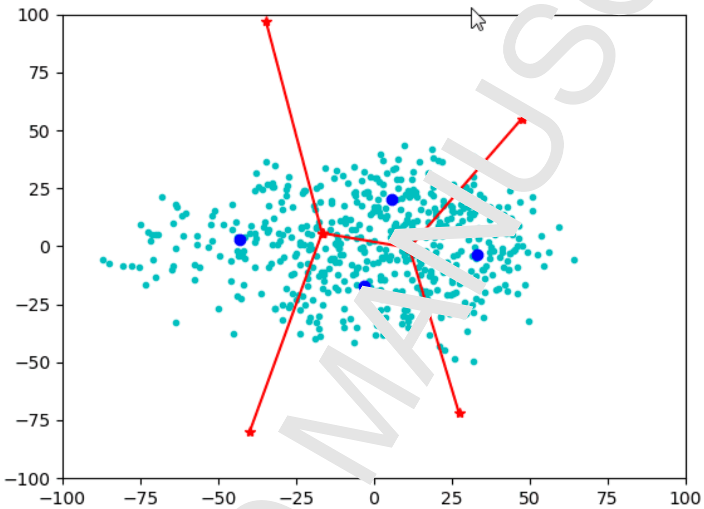


Figure 6: Centroidal Voronoi tessellation clustering.

The VD is a great distance-based strategy of space division in computational geometry. It divides the space into different non-overlapping polygon regions according to the number of given non-coincident seed nodes. There is one and only one seed node in every region, and the seed node is the nearest choice to all planar points in each single region than any other seed nodes. The ways to calculate VD are various, such as the grappa tree [37]. It is evolved from another data structure called link-cut tree that proposed by Sleator and Tarjan. It extends the given binary tree so that each original node has three linked nodes. By inserting an additional node to every node that lack of child and add a parent node for the root node, all original nodes on a tree have three nodes connected to it. In the extended tree, the new root node and leaf nodes are all external nodes. It performs well on query operation of VD with the first-order linear complexity at the algorithm level in O(log n) time.

5. Evaluation and Findings

In order to verify the performance of the proposed data indexing structures for range query processing in WSN, simulation experiments on real data have been implemented and the results shown so far are presented and analyzed in this section. In the follows, we first describe the experimental environments. Then, the experiments are quantitatively and qualitatively explored.

5.1. SIMULATION SETUP

A simulation prototype was implemented in Matlab. The experimental parameters of the energy model are summarized in Table 1. All simulations documented here are run on a Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz computer configured with 8 GB RAM and having Windows 10 (64 bits) as operating system.

Table 1: System Parameters and Setting

Parameter	Setting
Number of sensor nodes	500
Message size	8 bytes
Transmission distance	50m
Energy cost for radio transmitting a message	19.2uJ
Energy cost for radio receiving a message	3.2uJ
Energy cost for sensing a light intensity	100nJ
Energy cost in radio sleeping	0.016mW
Initial energy budget at each sensor node	1J

5.2. THEORETICAL PERFORMANCE ANALYSIS

Since search paths have O(log n) nodes in 1D range tree, these O(log n) subsets can be found in O(log n) time, which means answering range queries in

ACCEPTED MANUSCRIPT

O(log n) time. Storing sizes of the sets at nodes needs O(n) space while kd-tree also needs to store an O(n) space which responses 2D range query in worst-case time $O(\sqrt{n}+k)$, where k is the output size. Without loss of generality, the 3D range search complexity can try and then be deduced. For d-dim. range query, the space complexity of kd-tree is an O($d \cdot n$) space and the worst-case time complexity is Q($n^{1-1/d}$+m). By simplification of fractional cascading methods, for 2D range search, the final query time complexity is O(log n + k), while space is O($n \cdot logn$). Hence, a set of n points in the plane can be responded in O(n$log^{(d-1)}$ n) time into kd-tree of O(n$log^{(d-1)}$ n) size so that any d-dimensional range query takes O($log^{(d-1)}$ n + k) time, where k is the output size.

The distribution of the particles in the bounding box decides the quad-tree's complexity. The quad-tree is one of the tree-like hierarchical structure that is gradually divided from top to bottom, and every node contains at most four child nodes. It is suited to two-dimensional spatial data, because the given range of space is recursively divided into four equal subspaces until the depth of the tree reaches a defined threshold or meets a planned requirement. The structure of a quad-tree is not complicated so that it is easy to search and insert a data node when the spatial data objects are distributed uniformly. However, there may be a much deeper level of the quad-tree and the great waste of storage space if the distribution of the spatial data is not evenly, which makes low query efficiency. The complexity of inserting all the nodes is O(n log n)=O($n \cdot b$). (Since the max value of the distinct particles is 2^b, and then $logn \leq b$).

Before learning some algorithms solving the point-location queries problem, we lay the emphasis on the parameters of the clustering algorithms in which n is the number of nodes and k is the number of clusters. The first algorithm is k-means clustering algorithm whose time complexity is O($n \cdot k$) because of the complexity of the mathematical model. The second is more efficient and superior whose time complexity is O($n \cdot logn$). Unfortunately, the algorithms are difficult to understand using computational geometry. But later a algorithm called plane sweep was invented by Steven Fortune, whose time complexity is similar to the former one but easier to understand. Finally the most efficient algorithm called incremental algorithm was invented, the time complexity is O(log n).

5.3. IMPLEMENTATION AND PERFORMANCE EVALUATION

To realize a more efficient query processing, a hierarchical index structure is constructed. The distributed index tress then drives efficient processing of queries and imposes restrictions on the number of sensor points involved. For queries whose results have already been stored in the index structure, the results can be acquired by accessing one or some index nodes rather than numerous sensor nodes. VD data indexing algorithm has proven to perform well with regard to the latency and communication cost of a great variety of queries. The selection criterions may cover the following several metrics, such as query responding delay, energy consumption, as well as average network traffic. Specifically, the network traffic refers to the average number of messages forwarded and sent by all sensors, and it can greatly affect energy efficiency, which is the reason it is taken as the criterion for performance evaluation. The query responding delay refers to the time for query responding from the issuing of the query till the user's receiving of results. However, in our simulation, we have not taken the computation delay of sensor nodes into consideration, and the query responding delay is evaluated by the number of hops that lead to the longest path to trigger a query and receive the feedback.

The aggregated data (max, minimum, and average) needs to be calculated by each attribute of each sensor node on a periodical basis. And an update interval is specified by the administrator as much larger than the sensing interval. After each update interval ends, the aggregation including min, max, and average values of that interval, is sent by one node to its parent node within the index structure. If the sensing interval, for example, is set as 10 minutes, and the update interval of the index is set as 2 hours. Given different number of cluster levels in WSN, we can demonstrate how the increase in cluster levels lead to the reduction of energy cost in WSN. The following Figure 7 has illustrated the decrease of energy consumption goes along with the increase of number of levels in the hierarchy.

Image the case to process 1000 queries during 72 hours, as has shown in Figure 7. The location condition in a query determines such parameter. It is clear to witness an obvious huge increase in network traffic on load, along with the increase of involved node percentage. This is because all the involved nodes are supposed to report results. We then have made a comparison between the four data indexing methods, and under the circumstance that the query region is flooded by query node, and corresponding data are sent back to the query node by all sensor nodes that have query conditions satisfied. In order to evaluate the proposed multi data indexing methods, 1000 queries have been performed. As presented in Figure 8, the accumulative total network traffic is less for the VD with data indexing scheme than the other three schemes, due to the fact that query optimization has avoided the repeated access to the same data that are shared by multi queries. Moreover, the more the queries are, the more energy the multi query optimization can decrease, since index structure have already saved more results.

As presented in Figure 9, it is implied that the larger the network size is, the longer the query responding delay will be. This is attributed to the fact which the length of paths increases along with the WSN size when it comes to the sending of queries and receiving of results. Compared to the other data indexing methods, VD manages to realize a shorter delay. The main cause is the index structure can help it acquire partial or all results and it has no requirements to search all satisfied sensor nodes. To conclude, VD data indexing structure is suitable to be applied for large-scale networks, given its quick and energy-efficient processing of spatial range data query.

6. Concluding Remarks

IoT application will increasing as our society becomes more digitalized, for example in industry 4.0 and beyond. Hence, we need approaches that allow us to achieve low cost data sensing, collecting and processing, as well as aggregation.

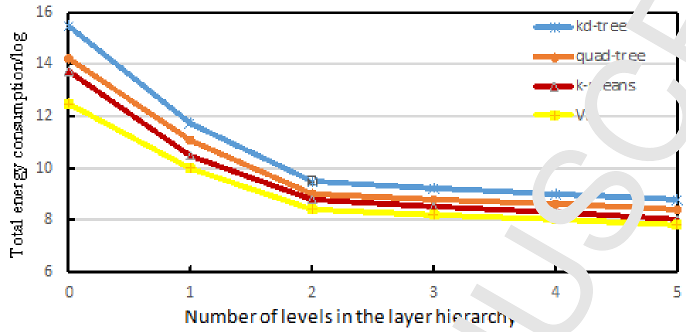


Figure 7: Total Energy consumption vs. number of levels in the layer hierarchy.

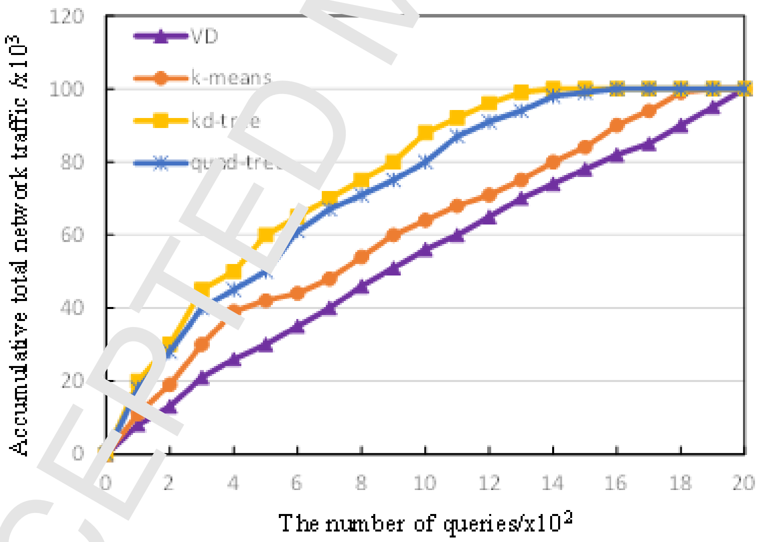


Figure 8: Accumulative network traffic of data indexing structures with multi query optimization strategy.

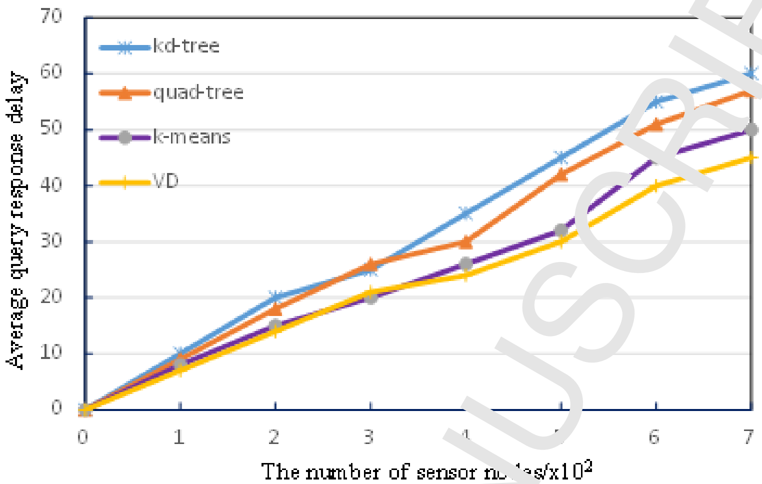


Figure 9: Query responding delay of variant sensor nodes in WSN.

In this paper, we proposed an architecture for distributed data indexing and evaluated its utility using simulations. There are, however, limitations in using simulations in the evaluation. Hence, one possible extension of this work is to implement a prototype of the proposed architecture, in collaboration with a real-world service provider. This will allow us to evaluate its utility in a real-world environment.

7. Acknowledgments

The authors thank the anonymous reviewers for their insightful suggestions. This work is supported by the National Natural Science Foundation of China under Grant No.61572454.

References

[1] S. R. Madden, M. J. Franklin, J. M. Hellerstein, W. Hong, Tinydb: an acquisitional query processing system for sensor networks, ACM Transactions on database systems (TODS) 30 (1) (2005) 122–173.

[2] W. Shu, X. Liu, Z. Gu, S. Gopalakrishnan, Optimal sampling rate assignment with dynamic route selection for real-time wireless sensor networks, in: Real-Time Systems Symposium, 2008, IEEE, 2008, pp. 431–441.

[3] L. Rao, X. Liu, Z. Gu, W. Liu, Hrs: A hierarchical routing and scheduling scheme for distributed real-time and embedded systems., Ad Hoc & Sensor Wireless Networks 11 (3-4) (2011) 265–284.

[4] S. Wan, Y. Zhang, Coverage hole bypassing in wireless sensor networks, The Computer Journal 60 (10) (2017) 1536–1544.

[5] S. Wan, Y. Zhang, J. Chen, On the construction of data aggregation tree with maximizing lifetime in large-scale wireless sensor networks, IEEE Sensors Journal 16 (20) (2016) 7433–7440.

[6] S. Wan, Energy-efficient adaptive routing and context-aware lifetime maximization in wireless sensor networks, International Journal of Distributed Sensor Networks 10 (11) (2014) 321964.

[7] R. Chen, A. P. Speer, M. Eltoweissy, Adaptive fault-tolerant qos control algorithms for maximizing system lifetime of query-based wireless sensor networks, IEEE Transactions on Dependable and Secure Computing 8 (2) (2011) 161–176.

[8] J. Niu, Z. Ming, M. Qiu, H. Su, Z. Gu, X. Qin, Defending jamming attack in wide-area monitoring system for smart grid, Telecommunication Systems 60 (1) (2015) 159–167.

[9] J. Wang, M. Qiu, B. Guo, Y. Shen, Q. Li, Low-power sensor polling for context-aware services on smartphones, in: High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on, IEEE, 2015, pp. 617–622.

[10] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, M. Qiu, A scalable and quick-response software defined vehicular network assisted by mobile edge computing, IEEE Communications Magazine 55 (7) (2017) 94–100.

[11] J. Tang, B. Zhang, Y. Zhou, L. Wang, An energy-aware spatial index tree for multi-region attribute query aggregation processing in wireless sensor networks, IEEE Access 5 (2017) 2080–2095.

[12] Z. Zhou, D. Zhao, X. Xu, C. Du, H. Sun, Periodic query optimization leveraging popularity-based caching in wireless sensor networks for industrial iot applications, Mobile Networks and Applications 20 (2) (2015) 124–136.

[13] R. I. Da Silva, D. F. Macedo, J. M. S. Nogueira, Spatial query processing in wireless sensor networks–a survey, Information Fusion 15 (2014) 32–43.

[14] J. Wen, J. Wang, Q. Zhang, Nearly optimal bounds for orthogonal least squares, IEEE Trans. Signal Process 65 (20) (2017) 5347–5356.

[15] S. Yang, M. A. Cheema, X. Lin, Y. Zhang, W. Zhang, Reverse k nearest neighbors queries and spatial reverse top-k queries, The VLDB Journal 26 (2) (2017) 151–176.

[16] D. Li, C. Zhang, J. Wen, A note on compact finite difference method for reaction–diffusion equations with delay, Applied Mathematical Modelling 39 (5-6) (2015) 1749–1754.

[17] M. Chen, Y. Zhang, M. Qiu, N. Guizani, Y. Hao, Spha: Smart personal health advisor based on deep analytics, IEEE Communications Magazine 56 (3) (2018) 164–169.

[18] J. Wen, B. Zhou, W. H. Mow, X.-W. Chang, An efficient algorithm for optimally solving a shortest vector problem in compute-and-forward protocol design, arXiv preprint arXiv:1410.4278.

[19] L. Yang, S. Li, Z. Xiong, M. Qiu, Hht-based security enhancement approach with low overhead for coding-based reprogramming protocols in wireless sensor networks, Journal of Signal Processing Systems 89 (1) (2017) 13–25.

[20] K. Wang, X. Hu, H. Li, P. Li, D. Zeng, S. Guo, A survey on energy internet communications for sustainability, IEEE Transactions on Sustainable Computing 2 (3) (2017) 231–254.

[21] I. Lazaridis, Q. Han, X. Yu, S. Mehrotra, N. Venkatasubramanian, D. V. Kalashnikov, W. Yang, Quasar: quality aware sensing architecture, ACM SIGMOD Record 33 (1) (2004) 26–31.

[22] H. Jiang, K. Wang, Y. Wang, M. Gao, Y. Zhang, Energy big data: A survey, IEEE Access 4 (2016) 3844–3861.

[23] C. Zhu, L. T. Yang, L. Shu, V. C. Leung, T. Hara, S. Nishio, Insights of top-k query in duty-cycled wireless sensor networks, IEEE Transactions on Industrial Electronics 62 (2) (2015) 1317–1328.

[24] J. Li, M. Qiu, Z. Ming, G. Quan, X. Qin, Z. Gu, Online optimization for scheduling preemptable tasks on IaaS cloud systems, Journal of Parallel and Distributed Computing 72 (5) (2012) 666–677.

[25] K. Gai, M. Qiu, Z. Ming, H. Zhao, L. Qiu, Spoofing-jamming attack strategy using optimal power distributions in wireless smart grid networks, IEEE Transactions on Smart Grid 8 (5) (2017) 2431–2439.

[26] D. Li, J. Zhang, Efficient implementation to numerically solve the nonlinear time fractional parabolic problems on unbounded spatial domain, Journal of Computational Physics 322 (2016) 415–428.

[27] G. Degirmenci, J. P. Kharoufeh, O. A. Prokopyev, Maximizing the lifetime of query-based wireless sensor networks, ACM Transactions on Sensor Networks (TOSN) 10 (4) (2014) 56.

[28] B. Chen, W. Liang, J. X. Yu, Energy-efficient top-k query evaluation and maintenance in wireless sensor networks, Wireless networks 20 (4) (2014) 591–610.

[29] H. Van Le, Distributed moving objects database based on key-value stores., in: PhD@ VLDB, 2016.

[30] Z. Ding, B. Yang, R. H. Güting, Y. Li, Network-matched trajectory-based moving-object database: Models and applications., IEEE Trans. Intelligent Transportation Systems 16 (4) (2015) 1918–1928.

[31] S. Alamri, D. Taniar, M. Safar, A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems 37 (2014) 232–242.

[32] E. R. Montiel, M. E. Rivero-Angeles, G. Rubino, H. Molina-Lozano, R. Menchaca-Mendez, R. Menchaca-Mendez, Performance analysis of cluster formation in wireless sensor networks, Sensors 17 (12) (2017) 2902.

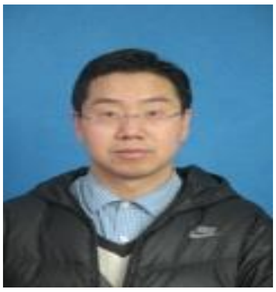
[33] M. Kulin, C. Fortuna, E. De Poorter, D. Deschrijver, I. Moerman, Data-driven design of intelligent wireless networks: An overview and tutorial, Sensors 16 (6) (2016) 790.

[34] G. Jesus, A. Casimiro, A. Oliveira, A survey on data quality for dependable monitoring in wireless sensor networks, Sensors 17 (9) (2017) 2010.

[35] D. M. S. Bhatti, N. Saeed, H. Nam, Fuzzy c-means clustering and energy efficient cluster head selection for cooperative sensor network, Sensors 16 (9) (2016) 1459.

[36] Y. Zhang, J. Wang, D. Han, H. Wu, R. Zhou, Fuzzy-logic based distributed energy-efficient clustering algorithm for wireless sensor networks, Sensors 17 (7) (2017) 1554.

[37] S. R. Allen, L. Barba, J. Iacono, S. Langerman, Incremental voronoi diagrams, Discrete & Computational Geometry 58 (4) (2017) 822–848.



Shaohua Wan received his joint Ph.D. degree from School of Computer, Wuhan University and Department of Electrical Engineering and Computer Science, Northwestern University, USA. From 2015, he worked as a postdoc at State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology. From 2016 to 2017, he worked as a visiting scholar at Department of Electrical and Computer Engineering at the Technical University of Munich. At present, he is an associate professor and master advisor at school of Information and Safety Engineering, Zhongnan University of Economics and Law. His main research interests include massive data computing for sensor networks and Internet of Things and edge computing.

Yu Zhao is a third-year Ph.D. Student of Computer Science at Technische Universit\"at M\"unchen (TUM). He is working in Image-Based Biomedical Modeling (IBBM) group with Prof. Bjoern H. Menze. Prior to coming to TUM, he received a Msc degree in signal and information processing and a BSc. degree in physics both from Beihang University (BUAA). His research focuses on the medical computer vision and application of machine learning, in particular medical image segmentation and high-level vision tasks.

Tian Wang received the B.Sc. and M.Sc. degrees in computer science from the Central South
University, Changsha, China, in 2004 and 2007, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Kowloon, Hong Kong, SAR, in 2011. He was a research assistant in City University of Hong Kong from 2006-2008. He is currently a Professor with the College of Computer Science and Technology, Huaqiao University, Xiamen, China. His research interests include Cyber Physical System, Cloud Computing and Fog Computing. Prof. Wang manages several research projects such as the National Natural Science Foundation of China (NSFC). He has 5 patents and more than 100 technical publications in international conferences and journals. His papers have appeared in the prestigious journals/conferences in the domain, including IEEE TMC, IEEE TVT, IEEE TOSC, ACM TOSN, ACM TETCI, Information Sciences, Computer networks, ACM Mobihoc, IEEE RTSS, IEEE MASS, IEEE ICC, and so on. He has served as publicity chair and program committee member of numerous international conferences. He serves as a general chair for IEEE SCS 2017, a publicity chair for IEEE DependSys 2016, session chair for SpaCCS 2016, track co-chair for IEEE CSS2017, and program committee member of numerous international conferences (SPCCIC 2014, APSCC 2014, HPCC 2015, CoCoNet'15, ICA3PP 2015, WASA 2015, HPCC 2016, DependSys 2015, DependSys 2016). He is an Associate Editor for the International Journal of Computers and Applications (Taylor & Francis), a Guest Editors for the journal of Cluster Computing and Concurrency and Computation: Practice and Experience. He is also on the

editorial board of International Journal of High Performance Computing and Networking (IJHPCN).

Zonghua Gu received his Ph.D. degree in Computer Science and Engineering from the University of Michigan at Ann Arbor under the supervision of Prof. Kang G. Shin in 2004. He worked as a post-doctoral researcher at the University of Virginia in 2004-05, and then as an assistant professor in at the Hong Kong University of Science and Technology in 2005-09 before joining Zhejiang University as an associate professor in 2009. His research area is real-time and embedded systems. He serves on the editorial board of the Journal of Systems Architecture (Elsevier)

QAMMER H. ABBASI (S'08–M'12–SM'16) received the B.Sc. and M.Sc. degrees (Hons.)in electronics and telecommunication engineering from the University of Engineering and Technology (UET), Lahore, Pakistan, and the Ph.D. degree in electronic and electrical engineering from the Queen Mary University of London (QMUL), U.K., in 2012. In 2012, he was a Post-Doctoral Research Assistant with the Antenna and Electromagnetics Group, QMUL. From 2012 to 2013, he was an International Young Scientist under National Science Foundation China and an Assistant Professor with UET. He is currently a Lecturer (Assistant Professor) with the School of Engineering, University of Glasgow. His research interests include nano communication, RF design and radio propagation, biomedical applications of millimeter and terahertz communication, wearable and flexible sensors, compact antenna design, antenna interaction with human body, Implants, body centric wireless communication issues, wireless body sensor networks, non-invasive health care solutions, physical layer security for wearable/implant communication, and multipleinput-multiple-output systems.

Kim-Kwang Raymond Choo holds the Cloud Technology Endowed Professorship in the Department of Information Systems and Cyber Security at the University of Texas at San Antonio, and is an Adjunct Associate Professor of Cyber Security and Forensics at the University of South Australia, Australia. He serves on the editorial board of Computers & Electrical Engineering, Cluster Computing, Digital Investigation, IEEE Access, IEEE Cloud Computing, IEEE Communications Magazine, Future Generation Computer Systems, Journal of Network and Computer Applications, PLoS ONE, Soft Computing, etc. He also serves as the Special Issue Guest Editor of ACM Transactions on Embedded Computing Systems (2017; DOI: 10.1145/3015662), ACM Transactions on Internet

Technology (2016; DOI: 10.1145/3013520), Digital Investigation (2016; DOI: 10.1016/j.diin.2016.08.003), Future Generation Computer Systems (2016; DOI: 10.1016/j.future.2016.04.017, 2018; DOI: 10.1016/j.future.2017.09.014), IEEE Cloud Computing (2015; DOI: 10.1109/MCC.2015.84), IEEE Network (2016; DOI: 10.1109/MNET.2016.7764272), IEEE Transactions on Cloud Computing (2017; DOI: 10.1109/TCC.2016.258[illegible]78), IEEE Transactions on Dependable and Secure Computing (2017; DOI: 10.1109/TDSC.2017.26[illegible]183), Journal of Computer and System Sciences (2017; DOI: 10.1016/j.jcss.2016.09.[illegible]01), Multimedia Tools and Applications (2017; DOI: 10.1007/s11042-016-4081-z), Personal and Ubiquitous Computing (2017; DOI: 10.1007/s00779-017-1043-z), Pervasive and Mobile Computing (2016; DOI: 10.1016/j.pmcj.2016.10.003), Wireless Personal Communications (2017; DOI: 10.1007/s11277-017-4278-0) etc. He is a Fellow of the Australian Computer Society, and a Senior Member of IEEE.

In this paper, in order to efficiently optimize the use of the network resources and improve the performance of energy consumption and query response time in WSNs, we propose a novel range data aggregation approach by exploiting spatial structures of sensory data. The contributions of this paper are summarized as follows:

We propose effective multidimensional data indexing structures to help process spatial queries efficiently, which provides a high-dimensional data indexing architecture for tackling the problems and enables us to present approaches which have much more applicable to mobility and spatial continuous range query than those proposed in previous works. In this scheme, the indexing scheme equally handles both types of information, and aggregates them in an energy efficient manner, providing a hierarchical in-network storage that is capable of timely responding to different queries,and further able to provide immediate answers to approximate queries and some types of exact queries.

In order to prove that the data indexing algorithms are generic enough to fit a wide variety of the commonly used spatial query processing,we present the applicability of the algorithm on four data structures:kd-tree, quad-tree, k-means clustering and Voronoi diagram (VD). VD data indexing model is suitable to general queries operations, which can, for example, be applied to process location-based service in the cells in O(log n) time.

Robust performance analysis is performed for the effect of each data structure in the data indexing. Our simulation results show the efficiency of the presented algorithm, in respect of query response time, and maintenance energy cost.