Short communication

Location-based grid-index for spatial query processing

Kwangjin Park*

Department of Electrical Information Communication Engineering, Wonkwang University, Iksan, Chunbuk, Republic of Korea

A R T I C L E   I N F O

Keywords:
Moving objects
Mobile computing
Spatial index

A B S T R A C T

Researchers have studied the various index structures and query processing algorithms to support efficient spatial queries in wireless communication environments. However, most previous studies have focused on server-side methods. In a server-side method, the server executes queries from individual clients and then relays the results. In our proposed method, the server transmits an index called DGI (Distributed Grid Index), and the client examines the received index to process spatial queries. The proposed index structure and search algorithm support efficient spatial queries in a wireless broadcast environment, shortening query search times.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

As mobile devices such as smartphones have become common, consumers more frequently use their mobile devices to communicate their real-time locations to others via diverse location-based services (LBSs) (Kang, Mokbel, Shekhar, Xia, & Zhang, 2010; Moon, Park, Chung, Sohn, & Ryu, 2005; Park & Valduriez, 2011). Social networking services (SNS), such as Twitter and Facebook, have aided the growth of these uses, allowing consumers to reveal their location during emergencies, learn about nearby restaurants and friends, and acquire local traffic and weather information. According to a survey conducted by Mobile Marketing Association and Luth Research, out of a group of 10,000 American adults, 63% of iPhone users in the United States use LBSs on their mobile phones at least once a week. Given the current growth of LBSs among cell-phone users, companies are boosting their efforts to automatically locate consumers for advertising and marketing purposes. Before smartphones, LBSs were considered hardware services. Now, LBSs are considered software services for mobile applications that incorporate user information and augment reality. These opportunities in software services represent a new business sector.

In general, the index provides an environment in which users who wish to search the database are shown only partial information so that they find the desired result more efficiently without having to go through the whole database one by one from the beginning to the end. Spatial index is used by spatial databases to optimize spatial queries (Galdames & Cai, 2012; Park, Song, & Hwang, 2004; Wang, Xu, Gu, Chen, & Yu, 2013).

The (1,m) index (Imielinski, Viswanathan, & Badrinath, 1997) is a well known air index techniques. In this method, the index is broadcast m times during a single broadcast cycle. The broadcast index is broadcast every fraction 1/m of the broadcast cycle (Park & Valduriez, 2011). The (1,m) index is suitable for a mobile computing environment, since less power is required under this technique.

Fig. 1 shows an example of expressing objects located in a two-dimensional space such as an R-tree index, a representative spatial index scheme. As shown in the figure, the root node is the total area and the A and B nodes represent the left and right areas of the root, respectively. The last node on the tree represents objects.

Researchers are beginning to study the use of LBSs in broadcast environments to effectively convey information to a group of general users (Park & Choo, 2010; Shanmugasundaram, Nithrakashyap, Sivasankaran, & Ramamritham, 1999; Zheng et al., 2003). In these studies, the server aggregates pieces of information intended for multiple users and then transmits the information, along with a spatial index, through a wireless broadcast channel. For example, using this method in a crowded park, a server can deliver information about coupons and menus from nearby restaurants and coffee shops to randomly selected members of LBSs. However, because battery power, signal speed, and computing power for handheld terminals have recently reached their limits, an efficient support system for processing spatial queries is needed. Spatial indices can shorten the tuning time of handheld terminals and expedite information searches. That is, the user's handheld terminal can retrieve the spatial index transmitted by the server prior to the spatial data, determine the availability of the desired information and the time of transmission, and selectively retrieve relevant sections of the data.

Indexes can be classified between ‘disk-based’ and ‘air’ indexes (Imielinski et al., 1997; Ku, Zimmermann, & Wang, 2008; Park & Choo, 2007). Generally, a disk-based index is used in point-to-point methods, where the server receives and processes the clients’
requests in an on-demand environment. In other words, the server searches for the query position and objects that are requested and then transfers the results to the requesting client. Contrastingly, an air-index is used in the push method, where the server transfers the broadcast programming information and the actual broadcast contents in sequence via wireless broadcast channels and the clients use the received index data to selectively tune to the necessary broadcast. Studies regarding spatial query processing in an air-index environment that can simultaneously handle requests from many users and also consider the privacy of the individual users are underway (Ku et al., 2008; Park, Choo, & Valduriez, 2010). Disk-based indexes have the following characteristics:

1. An index search at the time of the query request begins at the index starting point. Taking R-tree as an example, the index search always begins at the root, which is the topmost node.
2. Depending on the characteristics of the random-access media, the occurrence of backtracking during the index search does not have a significant effect on the search time.
3. A partial search of the index is possible. Taking R-tree as an example, a selective search is possible, which is limited to the potential locations of the query result.

Compared to disk-based indexes, air-indexes face three main issues: First, probe wait issue: Depending on the times of the query requests and the index transfers, the starting time for the index search can vary. Typically, the clients stand by at the index start point and then begin the index search. Second, backtracking issue: Backtracking of the index search with a tree structure has a significant impact on the query processing time. To hear again the past in the broadcast environment, it is necessary to wait until the next period. Third, partial index search issue: A partial search of the index is not possible. Repeating the awake and sleep modes and waiting until the desired data are transferred through the channel must be performed, according to the sequential approach.

Therefore, novel indexed structures and search algorithms should be devised to overcome the problems in wireless broadcasting environments and process queries efficiently. This paper proposes a new hierarchical distributed spatial index, called DGI (Distributed Grid Index), and a spatial query processing algorithm that exploits the objects’ locations and distribution data. The main contribution of this paper is with respect to the three main issues described above:

First, probe wait issue: The proposed index has the fully distributed non-redundant light-weight structure in order to reduce the probe wait time and to support the selective index search. Clients can determine the distance between the objects or the objects’ locations through the object identifications (IDs), which consist only of the location data of the objects. Therefore, an efficient data search is supported by a quick index search and short information retrieval cycle. Second, backtracking issue: The proposed index overcomes the problem of backtracking through the server, setting the broadcasting order with a consideration for the object’s location. Our Top–Down Search algorithm (TDS) provides fast query processing with minimum search costs by utilizing a location-based data transfer order that is transferred form the server. Third, partial index search issue: Our DGI utilizes the hierarchical object identification based on the location of the object so that the object’s geographical location can be determined just with partial index data. Therefore, a fast spatial query processing is possible through the search of the object’s location without the probe waiting regardless of when the client awakes. Furthermore, our performance evaluation shows the superiority of our solution compared to previous work related to air indexing.

In this paper, we propose a spatial index and query processing algorithm that takes the broadcast environment into consideration. The proposed index partitions the local area into grid-cells and assigns the unique ID number to each grid-cell. Searches in previous studies (Zheng et al., 2003; Lee & Zheng, 2005) were expensive because pointers and IDs were repeatedly transmitted during index structuring. The proposed index assigns numerical indices by location. Consequently, the search structure is lightweight and efficient. Furthermore, the proposed query processing algorithm transmits and searches object ID numbers sequentially based on objects’ locations, providing the client with optimal access time.

The main contributions of this paper can be summarized as follows:

1. We propose that all objects be assigned location identification numbers so that the locations of those objects may be identified.
2. We propose an index with a hierarchical structure and search algorithms to support selective tuning of clients and minimize energy consumption.
3. We conduct simulation experiments to validate the performance of our approach. The experimental results show that the DGI and the spatial query processing algorithm support more rapid and accurate queries as compared to conventional studies.

The remainder of the paper is organized as follows. Section 2 discusses background. Section 3 describes our DGI spatial index structure. Section 4 describes our spatial query processing algorithms. Section 5 gives our performance evaluation. Finally, Section 6 concludes.

2. Background

In a broadcast environment, the index structure has a large impact on the total search and retrieval time (Park & Valduriez, 2011; Park & Choo, 2010; Zheng et al., 2003). Conventional R-tree-based (Guttman, 1984) tree structures that support spatial objects are optimized for a disk-based environment and, thus, are inefficient in wireless broadcast environments. In these algorithms, the backtracking step involves searching a previously searched node. In a disk-based environment, the backtracking step significantly affects the search and retrieval time (Park & Valduriez, 2011; Lee & Zheng, 2005). Additionally, when the size of the location index increases, the amount of information that the server must transmit also increases. This size

---

1 The average duration for getting to the next index segment is called the probe wait (Imielinski et al., 1997).

2 A common problem with R-tree-based methods occurs at points that are covered by two or more sibling minimal boundary rectangles (MBRs). In this situation, the search method may need to explore several subtrees before the target object can be located. This will increase the search time.
increase extends the so-called ‘broadcast cycle’ and affects the client’s access time. Fig. 2(a) shows an MBR comprised of 6 objects based on R-tree rules. An MBR encompasses neighboring objects in a minimal rectangle. These MBRs (e.g., \( R_2, R_4, \) and \( R_6 \)) are then encompassed by a bigger MBR (e.g., \( R_1 \)). This MBR structure can be expressed in tree form, as shown in the hierarchical structure in Fig. 2(b). The leaf nodes of the tree are pointers to the objects in the tree.

We examine an example range query in an R-tree-based structure. The example is shown in Fig. 2(a) and (b). Shaded boxes in these figures represent the range query \( Q \).

Step (1) \( R_1 \), the root node, is searched.

Step (2) We examine the relationship between \( R_2 \) and \( Q \). \( R_2 \) is searched because \( R_2 \) intersects \( Q \), objects 1 and 2 are searched because they are located inside of \( R_2 \). Because object 2 is located inside the query range, object 2 is added to the query result. The current query result is 2.

Step (3) We examine the relationship between \( R_3 \) and \( Q \). \( R_3 \) is searched because \( R_3 \) intersects \( Q \), and objects 3 and 4 are searched because they are located inside of \( R_3 \). Because objects 3 and 4 are not located inside the query, the query result at the end of this step remains 2.

Step (4) We examine the relationship between \( R_4 \) and \( Q \). \( R_4 \) is not searched because \( R_4 \) does not intersect \( Q \). The query process is finished and the final query result, 2, is returned.

In summary, the search process for the R-tree-based structure is as follows: \( R_1 \rightarrow R_2 \rightarrow 1 \rightarrow 2 \rightarrow R_3 \rightarrow 3 \rightarrow 4 \rightarrow R_4 \rightarrow \) stop.

Next, we examine an example of a range query using a location-based information transmission structure, as shown in Fig. 2(c). The location-based information transmission structure determines the order in which object information is transmitted according to the location of the object. In the figure, an arrow indicates the order of the transmission sequence. The data are transmitted top-down with respect to the y-axis; i.e., sequentially from object 1 to object 2, object 2 to object 3, 3 to 4, 4 to 5, and 5 to 6.

In Lee and Zheng (2005), the authors propose a spatial index, called Distributed Spatial Index (DSI) to support location-based queries in wireless data broadcast systems. DSI has a distributed structure that mixes multiple search paths into a linear index structure which is distributed into the broadcast cycle. In Wang et al. (2013), authors address the problem of answering kNN, range, and RNN queries in road networks via broadcast channels. Then, they propose ISW-index for spatial queries in wireless broadcast environments. The ISW provides a pair of distance bounds, which is effective for pruning the search space. The search space is reduced by using ISW and subsequently the client can download as less as possible data for query processing, which can conserve the energy. In Galdames and Cai (2012), authors consider the problems of efficient processing of location-cloaked queries. They propose a generic model that can handle various types of queries, such as ranges queries and KNN queries, within a single unified platform. When the system is under loaded, a query is processed upon its arrival without any delay. On the other hand, when the system is overloaded, the queries pending in the queue are processed in batch. In Liu and Su (2013), authors investigate how to schedule the on-demand broadcast for the data with time constraints using multiple broadcast channels and provide two heuristics to schedule the data broadcast. They discuss the problem of how to generate a broadcast schedule on multiple channels. Then, the problem is formulated and referred to as the On-demand Broadcasting with Minimum Miss rate Problem. In Wang, Xiao, and Shu (2012), authors study the broadcast schedule problem for disseminating timely data to periodic continuous queries, and a systematic and efficient solution is provided. They present the first effort toward generating efficient broadcast schedules to achieve timing predictability for all admitted periodic continuous queries.

In a previous paper Park and Choo (2007), we proposed a novel broadcast-based spatial data dissemination and selective tuning scheme, namely ESS (Exponential Sequence Scheme), which provides clients with the ability to perform selective tuning and assists in reducing the client’s tuning time. The basic idea is to use exponential pointers from each data item. To the best of our knowledge, even until recently, DSI and ESS have been a representative study of index for supporting spatial query processing in wireless broadcast environments. However, they do not provide treatment of both the index structure and the data transfer schemes to improve query processing time.

As we mentioned in Section 2, the conventional R-tree-based index was developed for disk-based environments. In this context, backtracking forces the search process to wait for an initial broadcast cycle before the process can move backward to a higher node.

\[ \text{Fig. 2. Objects and index structure.} \]
in the tree. This delay increases the search time, particularly in wireless environments. The Hilbert–Curve index (HCI) (Zheng et al., 2003), which takes the broadcast environment into account, partially enables selective retrieval. HCI uses the Hilbert–Curve (HC) order. However, the HC order does not match the order of the data transmission one to one. Additionally, all contents of the particular data block must be retrieved, and this requirement is problematic. The distributed spatial index (DSI) (Lee & Zheng, 2005) approach also requires retrieving all data in a block. This requirement necessitates an additional mapping step to calculate the location of an object using its HC value. Assuming that \( n \) is the number of bits needed to represent the coordinates of an object, the conversion requires time \( O\left(n^2\right) \).

In this chapter, we introduce a Distributed Grid Index (DGI) that structures an index in the form of a hierarchical grid based on the location of the object. By eliminating unnecessary elements of information, minimizing the index size, and shortening the broadcast cycle transmitted from the server, the DGI approach can improve a client’s access time. Moreover, problems that are prone to occur in conventional tree structures, such as backtracking, are resolved through location-based sequential object transmission.

3. Location-based index

Unlike conventional approaches, the DGI approach constructs indices solely with the unique IDs of the objects on the map. As such, it removes unnecessary index structure information and provides an efficient search environment. To accomplish this, DGI uses unique, grid-based IDs to reference the locations of all objects. The initial zone is the topmost root grid. Assuming that the overall zone is a square, the zone is partitioned with a cross. This partition produces four equal root grids: \( R_0, R_1, R_2, \) and \( R_3 \), which are obtained by drawing a ‘\( z \)’ shape through the zone, starting from the upper-left corner. Additional partitions quadrisect existing partitions recursively using the same rule as the root grid. Partitioning continues to divide (split) nodes into child nodes until only one object exists in each child node. Finally, each grid is quadrisected as shown in Figs. 3(a) and (b), resulting in each node having its own unique ID. The number assigned to each partitioned zone is used as the unique ID for that corresponding grid. Zone partitions during the structuring of actual spatial indices are directly affected by the distribution of objects. All grids must individually satisfy the requirement that they contain no more than one object. For instance, consider Fig. 4, because \( R_0 \) only has one object, \( R_0 \) is not repartitioned. In contrast, although grids \( R_1 \) and \( R_2 \) both contain two objects, \( R_1 \) is partitioned into four grids and \( R_3 \) is partitioned into ten grids because of the location of their objects.

The number of digits in the unique ID of an object indicates the order of the grid. A unique ID with four digits indicates that the order of the grid is 4, which has a 4-level partition from the top grid 0 root to the grid-3 leaf, which refer to objects 1211 and 1212 of \( R_1 \), respectively, in Fig. 5. The DGI assigns every zone a differentiated order based on the distribution of objects. When a client is engaged in an object search, the client can confirm the location and the distribution of objects using only a given object’s unique ID.

In DGI, grid division and index ID allocation are assigned only to the areas where objects exist and, therefore, the size of index pointers is reduced, thereby shortening the index reading and tuning time. With regard to the delivery order, delivery is made in the order of the grids’ IDs in consideration of the locations of the objects; support is provided so that, only at a time point when information close to the query point comes, the client wakes up and processes the final query result. The client identifies information that the objects he or she intends to find exist within a certain grid and then identifies the information and coordinates to identify the information on the precise locations of the objects situated within the relevant grid.

We define the DGI as follows.

**Definition 1 (Distributed Grid Index).** In a given set of data regions, grid \( G_i \) repeats the hierarchical division process until only 1 object is included. The DGI tree has an unbalanced tree structure, and the leaf node grid contains a single object.
4. DGI-based spatial queries

In this chapter, we examine a location-based spatial query processing algorithm that utilizes the DGI. The spatial query processing algorithm moves from top to bottom relative to the given object. This approach provides an efficient query with the smallest possible index size. The server stores ID information, which is information on the locations of objects. The ID information is stored as a hierarchical tree structure according to the number, locations, and distributions of objects within a certain area. In addition, the data bucket, detailed information on objects a user intends to find (e.g., information on the prices of food at a restaurant and photos), is stored together. Thereafter, the ID information and data bucket are sequentially delivered through broadcast channels according to the locations of objects.

The details of the index search technique are as follows:

- The server transmits the unique ID number of the object. The unique ID depends on the grid order, not the coordinate location of the object. The order of transmission is from top to bottom with respect to the location of the object.
- The data transmits only the unique ID of the lowest-level grid, regardless of the hierarchical structure of the DGI grid. Therefore, the unique IDs of objects at various levels are transmitted once, and they depend on the locations and distribution of the objects.
- The client can make selective retrievals using the unique ID that was transmitted by the server.
- Because the data are transmitted sequentially based on the locations of the objects, the search algorithm can quickly obtain information without additional indexing.

We define an NN query processing using the DGI as follows.

**Definition 2 (Final NN decision for TDS).** While the data objects are sequentially broadcast in horizontal order, that is, from the top coordinate to the bottom coordinate, if the y-coordinate of the top of grid of \( G_{i+n} \) is lower than MMC, then \( G_{i+n} \) and the rest of the broadcast data objects are located outside of the NN range.

Let us explain Definition 2 with Fig. 4. If the y-coordinate of object is lower than the y-coordinate of bottom of MMC (e.g., object 4, object 5, object 6, object 7, and object 8), the object is located outside of the NN range.

Fig. 5. Grid partitioning according to the distribution of objects and the tree structure.

Fig. 6 shows pseudo-code for NN query processing based on DGI.

Let us describe the steps taken by the client to process the NN query with DGI.

- **Step (1)**: The ID number of object \( G_i \) is read, and the MMC is drawn with the grid that contains \( G_i \) and the query point as the radius. At the initial stage, \( n = 1 \); \( NN_{candidate}=\{G_i\} \), and MMC be MMC of \( G_i \).
- **Step (2)**: \( G_{i+n} \) is read. If \( G_{i+n} \) is EOF, then the coordinates of the NN candidate objects are read, and the final query results are determined. Otherwise, go to step (3).
- **Step (3)**: If the grid that contains object \( G_{i+n} \) is included in the MMC or is overlapped with MMC, then go to step (4); otherwise, go to step (5).
- **Step (4)**: NN-candidate \( \bigcup G_{i+n} \) (\( G_{i+n} \) can be the possible object for the final result of NN). Go to step (7).
- **Step (5)**: \( G_{i+n} \) cannot be the final result of NN. If the grid that contains object \( G_{i+n} \) is located at the coordinate below the MMC of the \( G_i \), go to step (6); otherwise, go to step (2).
- **Step (6)**: Stop tuning. In this case, \( G_{i+n} \) and all of the objects transferred afterwards cannot be the final query result for NN. Read the coordinates of the NN candidate objects, and determine the final query result.
- **Step (7)**: Draw the MMC of \( G_{i+n} \). If the MMC of \( G_{i+n} \) is smaller than the MMC of \( G_i \), then \( MMC <= MMC \) of \( G_{i+n} \); otherwise, \( MMC <= MMC \) of \( G_i \). Increase the value of \( n \). Go to step (2).

5. Simulation results

Even until recently, DSI and ESS have been a representative study of index for supporting spatial query processing in wireless broadcast environments. In this section, we evaluate the performance of DGI, (1, m) and DSS. DSS uses a distributed index that combines the characteristics of both DSI (Lee & Zheng, 2005) and ESS (Park & Choo, 2007) and each index table has exponential pointers. DSI and ESS are proposed recently to improve query processing that assumed the most similar environments as the proposed technique. In this section, we evaluated access latency for DGI, DSI (Lee & Zheng, 2005), and (1, m) index (Imielinski et al., 1997) with a Pentium 3.16 GHz CPU. We assume that the broadcast channel has a bandwidth of 2 Mbps (Mouratidis, Bakiras, & Papatdias, 2009, Park et al., 2004). We use a real dataset (hereafter called D1) containing 39,231 data objects of MBRs for Montgomery.

\[ \text{MMC} \] is drawn with the grid that contains \( G_i \) and the query point as the radius.

\[ \text{MMC} \] is located at the coordinate below of grid of \( D_i \).
County roads and a uniform dataset (hereafter called \(D_2\)) with 10,000 points; the former was extracted from the dataset available at http://www.rtreeportal.org. For default setting, we assign the 10 bytes for each pointer which also includes two dimensional coordinates. We assign 32 bytes for the spatial index since it needs more space to maintain spatial information and pointers in index.

Tables 1 and 2 show the access times for the DGI, the DSS, and the \((1,m)\) index resulting from an increase in the data size of an NN query in \(D_1\) and \(D_2\), respectively. As shown in the tables, the DGI has low access latency compared to the other index methods. This low latency is because the DGI supports object location ascertainment and selective retrieval with only the object ID; the DSS and \((1,m)\) methods require additional information.

Tables 3 and 4 show the access times for the DGI, the DSS, and the \((1,m)\) index resulting from an increase in the pointer size of an NN query process in \(D_1\) and \(D_2\), respectively. As shown in the tables, the DGI has a relatively low access latency as compared to the other index methods. The DGI has low access latency because it is not greatly affected by increases in the pointer size. In contrast, the DSS is affected greatly by increases in the pointer size because the number of pointers is related to the log of the total data size.

6. Conclusion

In this paper, we proposed a new index and a search algorithm that considers the spatial distribution of objects. The proposed DGI represents the location and distribution of objects using a grid cell. Consequently, unlike conventional index methods, the DGI can accurately ascertain the location of an object using only the object’s unique ID. Unique IDs are transmitted top–down from the object located at the highest point to the object located at the lowest point, given the locations of known objects on the map. The search algorithm supports selective spatial queries based on the order of transmission and the unique ID of a given object. The experimental results proved that the DGI and the spatial query processing algorithm support more rapid and accurate queries as compared to conventional studies.

Acknowledgment

This paper was supported by Wonkwang university in 2013.

References


