Location-based Spatial Queries with Data Sharing in Wireless Broadcast Environments *

Wei-Shinn Ku Roger Zimmermann
University of Southern California
Los Angeles, California 90089
[wku, rzimmerm]@usc.edu

Haixun Wang
IBM T.J. Watson Research Center
Hawthorne, New York 10532
[haixun]@us.ibm.com

Abstract

Location-based spatial queries (LBSQs) refer to spatial queries whose answers rely on the location of the inquirer. Efficient processing of LBSQs is of critical importance with the ever-increasing deployment and use of mobile technologies. We show that LBSQs have certain unique characteristics that traditional spatial query processing in centralized databases does not address. For example, a significant challenge is presented by wireless broadcasting environments, which often exhibit high-latency database access. In this paper, we present a novel query processing technique that, while maintaining high scalability and accuracy, manages to reduce the latency considerably in answering location-based spatial queries. Our approach is based on peer-to-peer sharing, which enables us to process queries without delay at a mobile host by using query results cached in its neighboring mobile peers. We illustrate the appeal of our technique through extensive simulation results.

1. Introduction

Spatial query processing is becoming an integral part of many new applications. Recently, there has been a growing interest in the use of location-based spatial queries (LBSQs), which refer to a set of spatial queries that retrieve information based on mobile users' current locations [10]. User mobility and data exchange through wireless communication give LBSQs some unique characteristics that traditional spatial query processing in centralized databases does not address. Novel query processing techniques must be devised to handle these new challenges.

- ▷ **Mobile Query Semantics.** In a mobile environment, a typical LBSQ is of the following form: "find the top-three nearest hospitals." The result of the query depends on the location of its requester. Caching and sharing of query results must take into consideration the location of the query issuer.
- ▷ **High Workload.** The database resides in a centralized server, which typically serves a large mobile user community through wireless communication. Consequently, bandwidth constraints and scalability become the most important design concerns of LBSQ algorithms.
- ▷ **Query Promptness and Accuracy.** Due to users' mobility, answers to an LBSQ will lose their relevancy if there is a long delay in query processing or in communication. For example, answers to the query "find the top-three nearest hospitals" received after five minutes of high-speed driving will become meaningless.

The wireless environment and the communication constraints play an important role in determining the strategy for processing LBSQs. In the simplest approach, a user establishes a point-to-point communication with the server so that her queries can be answered on demand. However, this approach suffers from several drawbacks. First, it may not scale to very large systems. Second, to communicate with the server, a client must most likely use a fee-based cellular-type network to achieve a reasonable operating range. And third, users must reveal their current location and send it to the server, which may be undesirable for privacy reasons. A more advanced solution is the wireless broadcast model. It can support an almost unlimited number of mobile hosts (MH) over a large geographical area with a single transmitter. With the broadcast model, mobile hosts do not submit queries – instead they tune in to the broadcast channel for information which they desire. Hence, the user's location is not revealed. One of the limitations of the broadcast model is that it restricts data access to be sequential. Queries can only be fulfilled after all the required on-air data arrives. This is why in some cases, a five-minute delay to the query "find the top-three nearest hospitals" would not be unusual.

Alleviating this limitation, we propose a scalable and low latency approach for processing location-based spatial queries in broadcast environments. Our approach leverages ad-hoc networks to share information among mobile clients in a peer-to-peer (P2P) manner. The rationale for our approach is based on the following observations.

- As mentioned previously, when a MH launches a nearest neighbor query, in many situations, she would prefer an approximate result that arrives with a short response time rather than an accurate result with a long latency.
- The results of spatial queries often exhibit spatial locality. For example, if two mobile hosts are close to each

*This research has been funded in part by NSF grants EEC- 9529152 (IMSC ERC) and IIS-0534761, and equipment gifts from Intel Corporation, Hewlett-Packard, Sun Microsystems and Raptor Networks Technology.

1-4244-0803-2/07/$20.00 ©2007 IEEE

other, the result sets of their spatial queries may overlap significantly. Query results of a mobile peer are valuable for two reasons: i) they can be used to answer queries of the current mobile host directly; and ii) they can be used to dramatically reduce the latency for the current mobile host to retrieve on-air information.

- P2P approaches can be valuable for applications where the response time is an important concern. Through mobile cooperative caching [2] of the result sets, query results can be efficiently shared among mobile clients.

In this paper, we concentrate on two common types of spatial searches, namely, k nearest neighbor queries and window queries. The contributions of our study are as follows.

a) We identify certain characteristics of LBSQs that enable the development of effective sharing methods in broadcast environments.

b) We utilize a P2P based sharing method to improve the current approaches in answering on-air k nearest neighbor queries and window queries.

c) Through extensive simulation experiments, we evaluate the benefits of our approach under different parameter sets.

2. Related Work

2.1 Wireless data broadcast

We can distinguish two approaches for mobile data access: the *on-demand access model* and the *wireless broadcast model*. For the on-demand access model, point-to-point connections are established between the server and the mobile clients, and the server processes queries which the clients submit on demand. For the wireless broadcast model, the server repeatedly broadcasts all the information in wireless channels and the clients are responsible for filtering the information. The advantage of the broadcast model is that it is a scalable approach. However, the broadcast model has a large latency, as clients have to wait for the information they need in a broadcasting cycle. If a client misses the packets which it needs, it has to wait for the next broadcast cycle. Nearly all the existing spatial access methods are for databases with random access disks. These existing techniques cannot be used in a wireless broadcast environment, where only sequential data access is supported. Zheng et al. [11] proposed to index the spatial data on the server by a space-filling curve. The Hilbert curve is chosen for this purpose because of its superior locality. The index values of the data packets represent the order in which these data packets are broadcast. For example, the Hilbert curve in Figure 1 tries to group data of close values so that they can be accessed within a short interval when they are broadcast sequentially. The mobile hosts use on-air search algorithms [11] to answer location-based spatial queries (k nearest neighbor and window queries) over data that arrives in the order prescribed by the Hilbert curve.

2.2 Spatial Queries and Cooperative Caching

We focus on two common types of spatial queries: k nearest neighbor queries and window queries. For nearest neighbor queries, the use of the R-tree algorithm [4] and its derivatives, and the branch-and-bound algorithms that search an R-tree in either a depth-first [7] or best-first manner [5], have been widely adopted. For window queries that find objects within a specified area, the R-tree families [8, 1] provide efficient access to disk-based databases. Basically, an R-tree groups objects close to each other into a minimum bounding rectangle (MBR), and a window query only visits the MBRs that overlap with the query window.

Caching is a key technique to improve data retrieval performance in widely distributed environments. With the increasing deployment of new P2P wireless communication technologies (e.g., IEEE 802.11b/g), *peer-to-peer cooperative caching* becomes an effective sharing alternative [9]. With this technique, mobile hosts communicate with neighboring peers in an *ad-hoc* manner for information sharing, instead of relying solely on the communication with remote information sources. Peer-to-peer cooperative caching can bring about several distinctive benefits to a mobile system: improved access latency, reduced server workload, and alleviated point-to-point channel congestion.

3. System Design

In this section, we describe our approach for supporting LBSQs in a wireless broadcast environment. The fundamental idea behind our methodology is to leverage the cached results from prior spatial queries at reachable mobile hosts for answering future queries at the local host.

The wireless data broadcast model has good scalability for supporting an almost unlimited number of clients [6]. Its main limitation lies in its sequential data access; the access latency becomes longer as the number of data items increases. If we can provide (approximate) answers to spatial queries before the arrival of the related data packets, we will overcome the limitation of the broadcast model. A novel component in our methodology is a verification algorithm that verifies whether a data item from neighboring peers is part of the solution set to a spatial query. Even if the verified results constitute only part of the solution set, in which case the query client needs to wait for the required data packets to get the remaining answers, the partial answer can be utilized by many applications that do not need exact solutions but require a short response time (for example, the query "What are the top 3 nearest hospitals?" issued by a motorist on a highway).

3.1 Assumed Infrastructure

Our operating environment contains two main entities: a remote wireless information server and mobile hosts. We are considering mobile clients, such as vehicles, that are instrumented with global positioning systems (GPS) for continuous position information. Furthermore, we assume that the

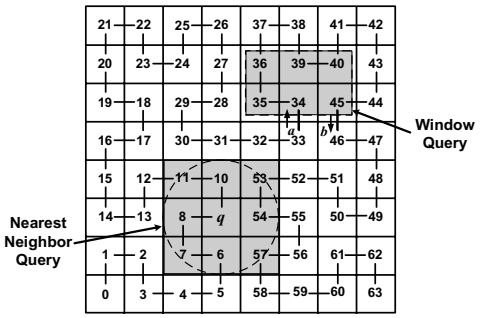

Figure 1. On air spatial query examples. The numbers represent index values.

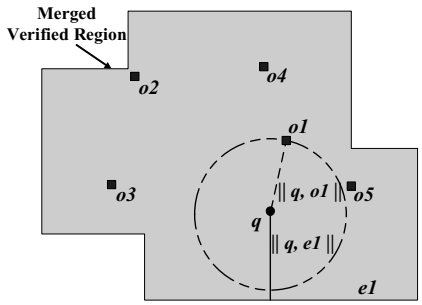

Figure 2. Because e_1 has the shortest distance to q and $||q, o_1|| \leq ||q, e_1||$, POI o_1 is verified as a valid NN of mobile host q.

wireless information server broadcasts information in a wireless channel periodically and the channel is open to the public. In addition, there are short-range networks that allow ad-hoc connections with neighboring mobile clients. Technologies that enable ad-hoc wide band communication include, for example, IEEE 802.11b/g. Benefiting from the power capacities of vehicles, we assume that each mobile host has a significant transmission range and virtually unlimited power lifetime. The architecture also supports hand-held mobile devices. When a mobile host q issues a spatial query, it tunes into the broadcast channel and waits for the data. In the meantime, q can collect cached spatial data from peers to harvest existing results in order to complete its own spatial query. Because memory space is scarce in mobile devices, we assume that each mobile host q caches a set of POIs in an MBR related to its current location. Since the POIs located inside the MBR are from the wireless information server, we define the area bounded by the MBR as a *verified region*, $q.VR$, with regard to q's location.

3.2 Sharing Based Nearest Neighbor Queries

Figure 1 shows an example of an on air kNN query based on a Hilbert curve index structure [11]. At first, by scanning the on air index, the k^{th} nearest object to the query point is found and a minimal circle centered at q and containing all those k objects is constructed. The MBR of that circle, enclosing at least k objects, serves as the search range. Consequently, q has to receive the data packets that covers the MBR from the broadcast channel for retrieving its k nearest objects. As shown in Figure 1, the related packets span a long segment in the index sequence – between 5 and 58 – which will require a long retrieval time. The other problem of this search algorithm is that the indexing information has to be replicated in the broadcast cycle to facilitate twice-scanning. The first scan is for deciding the kNN search range and the second scan is for retrieving k objects based on the search range [11].

Therefore, we propose a *sharing based nearest neighbor* (SBNN) query approach to improve the current on air kNN query algorithm. The SBNN algorithm attempts to verify the validity of k objects by processing results obtained from several peers. When a mobile host q executes SBNN, it first broadcasts a request to all its single-hop peers for their cached spatial data. Each peer that receives the request returns the verified region MBR and the cached points of interest to q. Then q combines the verified regions of all the replied peers, each bounded by its MBR, into a merged verified region M_{VR} (the polygon in Figure 2). The merging process is carried out by the *MapOverlay* algorithm [3] (line 4 of Algorithm 1). The core of SBNN is the *nearest neighbor verification* (NNV) method, whose objective is to verify whether a POI o_i obtained from peers is a valid (i.e., top k) nearest neighbor of the mobile host q.

Let P denote the data collected by q from j peers $p_1, \cdots, p_j$. Consequently, the merged verified region M_{VR} can be represented as:

$$M_{VR} = p_1.VR \cup p_2.VR \cup \cdots \cup p_j.VR.$$

Suppose the boundary of M_{VR} consists of k edges, $E = \{e_1, e_2, \ldots, e_k\}$ and there are l points of interest, $O = \{o_1, o_2, \ldots, o_l\}$, inside the M_{VR}. Let $e_s \in E$ be the edge that has the shortest distance to q. An example is given by Figure 2, where $k = 10$, and e_1 has the shortest distance to q.

Lemma 3.1 *Let $\widehat{O} = \{\widehat{o}_1, \widehat{o}_2, \ldots, \widehat{o}_v\}$ be a set of POIs each of which is closer to q than e_s and q is inside M_{VR}. Then, $\widehat{o}_1, \widehat{o}_2, \ldots, \widehat{o}_v$ are the top v nearest neighbors of q.*

Proof:

Assume o_m is one of the top v nearest neighbors of q, but $o_m \notin \widehat{O}$. Then, $||q, o_m|| < ||q, \widehat{o}_v||$ and $||q, o_m|| < ||q, e_s||$. Since $||q, o_m|| < ||q, e_s||$, o_m must be inside M_{VR} and $o_m \in O$. Based on the definition of $\widehat{O}$, o_m must be a member of $\widehat{O}$. However, this contradicts the assumption that $o_m \notin \widehat{O}$. Therefore, $\widehat{O}$ must cover the top v nearest neighbors of q. ∎

In Figure 2, according to Lemma 3.1, the POI o_1 can be verified as the nearest neighbor of q and is termed a *verified nearest neighbor*, because the Euclidean distance between o_1 and q is no greater than the Euclidean distance between e_1 and q.

The NNV method uses a heap H to maintain the entries of verified and unverified points of interest discovered so far. Initially H is empty. The NNV method adds POIs to H

as it verifies objects from mobile hosts in the vicinity of q. The heap H maintains the POIs in an ascending order in terms of their Euclidean distances to q. Unverified objects are kept in H only if the number of verified objects is lower than requested by the query. The nearest neighbor verification method is formalized in Algorithm 1. Since the verified region merging process dominates the algorithm complexity, the SBNN algorithm can be computed in $O(n \log n + i \log n)$ time, where n is the total number of edges of the two merged polygons and i is the number of intersection points.

3.3 Sharing Based Window Queries

As proposed in [11], the basic idea for a mobile host to process a window query w based on space-filling curve index is to decide a candidate set of points along the Hilbert curve. The candidate set includes all the points that fall within the query window of w. Then the MH retrieves the related packets and filters out data objects which are located outside of the query window. As illustrated in Figure 1, the dashed-line rectangle represents the query window w. We can find a first point a and a last point b according to the order in which they occur on the Hilbert curve. Consequently, all the points inside this query window should lie on the Hilbert curve segmented by points a and b.

Although the algorithm proposed in [11] can find entry and exit bounding points on a Hilbert curve index to decrease the number of candidate points, the *access latency* is still very long. As shown in the example, the required data packets span between index value 33 and 46 and cover around 20% of the whole data file. Although a search space partitioning technique was proposed in [11] for improving the performance, it still cannot mitigate the overhead of access latency. Therefore, we propose a *Sharing Based Window Query* (SBWQ) method to improve the current on air window query algorithm.

For SBWQ, The MH q first broadcasts a request to all its single-hop peers for requesting their cached spatial data.

Algorithm 1 SBNN (q, H, k)

1: $P \leftarrow$ peer nodes responding the query request issued from q.
2: $M_{VR} \leftarrow \emptyset$
3: **for** $\forall p \in P$ **do**
4: $\quad M_{VR} \cup = p.VR$ and $O \cup = p.O$
5: **end for**
6: $\forall o_i \in O$, sort according to $||q, o_i||$
7: Compute $||q, e_s||$ where edge e_s has the shortest distance to q among all the edges of M_{VR}
8: i = 1
9: **while** $|H| < k$ and $i \leq |O|$ **do**
10: $\quad$ **if** $||q, o_i|| \leq ||q, e_s||$ **then**
11: $\quad\quad H$.verified $\cup = o_i$
12: $\quad$ **else**
13: $\quad\quad H$.unverified $\cup = o_i$
14: $\quad\quad i$++
15: $\quad$ **end if**
16: **end while**
17: **return** H

Parameter	LA City	Riverside County	Synthetic Suburbia	Units
POI_{Number}	2750	1450	2100	
MH_{Number}	93300	9700	51500	
C_{Size}	50	50	50	
λ_{Query}	6220	650	3440	min^{-1}
Tx_{Range}	200	200	200	m
λ_{kNN}	5	5	5	
λ_{window}	3	3	3	%
$\lambda_{Distance}$	1	1	1	mile
$T_{execution}$	10	10	10	hr

Table 1. The simulation parameter sets.

Then it combines the returned verified regions $p.VR$, each bounded by its MBR, into a *merged verified region* M_{VR}. Next q computes the spatial relationship between the query window w and the merged verified region M_{VR}. If w can be totally covered by M_{VR}, the window query can be fulfilled. Otherwise, the whole or part of the query window must be solved as an on air window query. However, under these conditions we may be able to reduce the query window. The SBWQ algorithm is formalized in Algorithm 2.

4. Performance Evaluation

The main purpose of our peer-to-peer design is to decrease the access latency, as queries are answered directly by peers. Consequently, the focus of our simulation is to quantify what percentage of the client spatial query requests can be fulfilled by peers. We acquired our simulation parameters from real world data sets, for example vehicle and gas station densities in Southern California. We named the two parameter sets the *Los Angeles City* and the *Riverside County* parameter set. For comparison purposes we blended the two real parameter sets to generate a third, synthetic data set. The simulation results of window queries are omitted due to space limitations, however they exhibit a similar trend as the kNN results. Table 1 lists the three parameter sets.

4.1 Transmission Range Experiments

We first varied the mobile host wireless transmission range from 10 meters to 200 meters as demonstrated in Figure 3. As the transmission range extends, an increasing number of queries can be answered by surrounding peers. Because of its high vehicle density, the effect is most prominent with the Los Angeles City parameter set. With a 200 meter transmis-

Algorithm 2 SBWQ(q, w)

1: $P \leftarrow$ peer nodes responding the query request issued from q.
2: **for** $\forall p \in P$ **do**
3: $\quad M_{VR} \cup = p.VR$ and $O \cup = p.O$
4: **end for**
5: $WQ \leftarrow \forall o \in O$ which overlap with w
6: **if** $w \subset M_{VR}$ **then**
7: $\quad$ **return** WQ
8: **else**
9: $\quad WQ \cup$ query results returned from the on air window query with w'. {if $w \not\subset M_{VR}$, utilize w' to compute the new search bounds and results.}
10: $\quad$ **return** WQ
11: **end if**

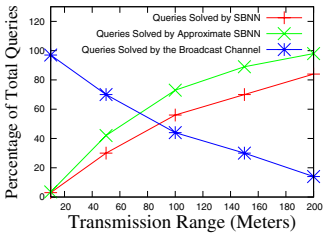

Fig. 3a. Los Angeles City.

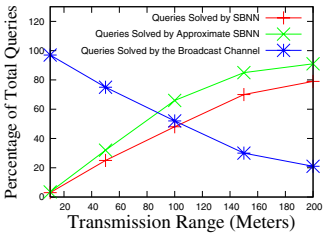

Fig. 3b. Synthetic Suburbia.

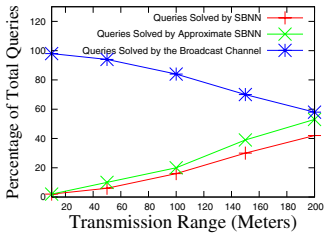

Fig. 3c. Riverside County.

Figure 3. The percentage of resolved queries as a function of the wireless transmission range.

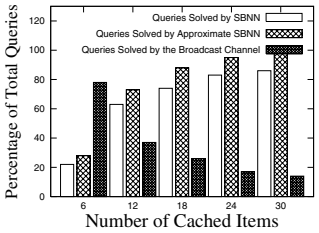

Fig. 4a. Los Angeles City.

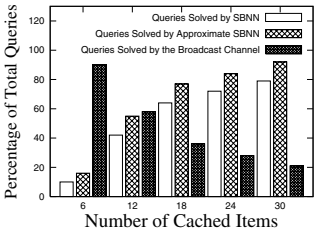

Fig. 4b. Synthetic Suburbia.

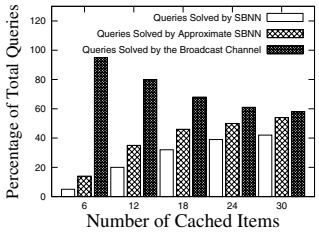

Fig. 4c. Riverside County.

Figure 4. The percentage of resolved queries as a function of the mobile host cache capacity.

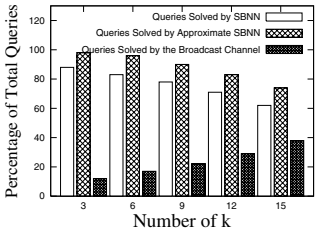

Fig. 5a. Los Angeles City.

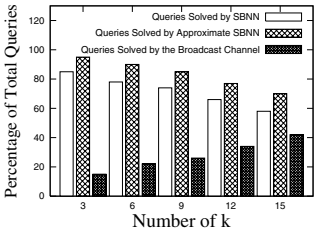

Fig. 5b. Synthetic Suburbia.

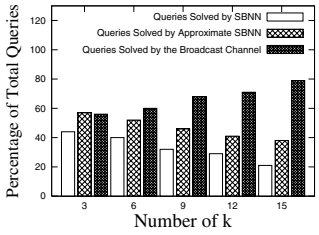

Fig. 5c. Riverside County.

Figure 5. The percentage of resolved queries as a function of k.

sion range, less than 20% of the queries must be solved by listening to the broadcast channel for exact results.

4.2 Cache Capacity Experiments

Figure 4 illustrates the increase of the cache capacity from 6 to 30 POI objects with the three parameter sets. Even though the total number of interest objects is much larger than the maximum cache capacity, we observe a remarkable increase of queries solved by SBNN with a higher mobile host cache capacity in Figures 4a and b.

4.3 Nearest Neighbor Number k Experiments

We altered k in the range from 3 to 15 as the mean number for each query. As shown in Figure 5, the solved queries by the broadcast channel for the LA City parameter set increased 28% when we raised k from 3 to 15. The solved queries for the Riverside County parameter set increased by only 21%, because its starting level was much higher. Not surprisingly our technique is much more effective for small values of k.

5. Conclusions

This paper presented a novel approach for reducing the spatial query access latency by leveraging results from nearby peers in wireless broadcast environments. Significantly, our scheme allows a mobile client to locally verify whether candidate objects received from peers are indeed part of its own spatial query result set. By virtue of its peer-to-peer architecture, the method exhibits great scalability: the higher the mobile peer density, the more queries can be answered by peers. Therefore, the query access latency can be markedly decreased with the increase of clients.

References

[1] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *SIGMOD Conference*, pages 322–331, 1990.

[2] C.-Y. Chow, H. V. Leong, and A. T. S. Chan. Distributed Group-based Cooperative Caching in a Mobile Broadcast Environment. In *Mobile Data Management*, pages 97–106, 2005.

[3] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications (2nd Edition)*. Springer, 2000.

[4] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD Conference*, pages 47–57, Boston, Massachusetts, June 18-21, 1984.

[5] G. R. Hjaltason and H. Samet. Distance Browsing in Spatial Databases. *ACM Trans. Database Syst.*, 24(2):265–318, 1999.

[6] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Data on Air: Organization and Access. *IEEE Trans. Knowl. Data Eng.*, 9(3):353–372, 1997.

[7] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest Neighbor Queries. In *SIGMOD Conference*, pages 71–79, 1995.

[8] T. K. Sellis, N. Roussopoulos, and C. Faloutsos. The R+-Tree: A Dynamic Index for Multi-Dimensional Objects. In *VLDB*, pages 507–518, 1987.

[9] L. Yin and G. Cao. Supporting cooperative caching in ad hoc networks. *IEEE Trans. Mob. Comput.*, 5(1):77–89, 2006.

[10] J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D. L. Lee. Location-based Spatial Queries. In *SIGMOD Conference*, pages 443–454, 2003.

[11] B. Zheng, W.-C. Lee, and D. L. Lee. Spatial Queries in Wireless Broadcast Systems. *Wireless Networks*, 10(6):723–736, 2004.