

# Location-based Spatial Queries with Data Sharing in Mobile Environments

Wei-Shinn Ku and Roger Zimmermann  
Computer Science Department  
University of Southern California  
Los Angeles, CA 90089  
[weishinn, rzimmerm]@cs.usc.edu

## Abstract

*Mobile clients feature increasingly sophisticated wireless networking support that enables real-time information exchange with remote databases. Location-dependent spatial queries, such as determining the proximity of stationary objects (e.g., restaurants and gas stations) are an important class of inquiries. We present novel approaches to support nearest-neighbor queries and window queries from mobile hosts by leveraging the sharing capabilities of wireless ad-hoc networks. We illustrate how previous query results cached in the local storage of neighboring mobile peers can be leveraged to either fully or partially compute and verify spatial queries at a local host. The feasibility and appeal of our technique is illustrated through extensive simulation results that indicate a considerable reduction of the query load on the remote database. Furthermore, the scalability of our approaches is excellent because a higher density of mobile hosts increases its effectiveness.*

## 1 Introduction

Location-based queries are of interest in a growing number of applications and two important sub-classes of such queries are nearest neighbor (NN) searches and window queries. Increasingly such queries are issued from mobile clients and there exist several algorithms that allow the efficient execution of NN queries and window queries on centralized databases. In my PhD study I propose an approach that leverages short-range, ad-hoc networks to share information [2] in a peer-to-peer (P2P) manner among mobile clients to answer location-based queries. Such a P2P sharing model can be very valuable for applications where access to the server is not always guaranteed and may be spurious at times. For example, during a natural disaster such as an earthquake or a hurricane, the communication from rescue crews to stationary databases may be intermittent. In such a scenario, P2P data sharing can provide a robust alternative where fault-resilience is naturally built into the de-

sign. In addition, the sharing model can also be utilized in wireless broadcast environments [8] to decrease the access latency.

The efficiency of my approach is derived from the observation that the results of spatial queries often exhibit spatial locality. For example, if two mobile hosts (MH) are close to each other, the result sets of their  $k$ NN queries for a specific object type may overlap significantly. Through mobile cooperative caching of the result sets, query results can be efficiently shared among mobile clients [3, 15]. The contributions of my study are as follows. I first identify a set of characteristics that enable the development of effective sharing methods. Then I introduce a set of algorithms that aid in the decision process within this distributed environment to verify whether the data items received from neighboring clients provide a complete, partial, or irrelevant answer to the posed spatial query. Through extensive simulation experiments I explore the benefits of my approach under different parameter sets (e.g., mobile host density, wireless transmission range) [9].

## 2 Related Work

The existing work relevant to our approach can be broadly classified into *spatial query processing* and *cache management in mobile environments*.

**Spatial Query Processing** R-trees [6] and their derivatives have been a prevalent method to index spatial data and increase query performance. To find nearest neighbors, branch-and-bound algorithms have been designed that search an R-tree in either a depth-first [12] or best-first manner [7].

The purpose of a window query is to find the objects which are within a given query window. The R-tree family [13, 1] provide efficient index techniques to solve window queries. Basically, an R-tree groups objects close to each other into a minimum bounding rectangle (MBR), and a window query only visits the MBRs which overlap with the query window.

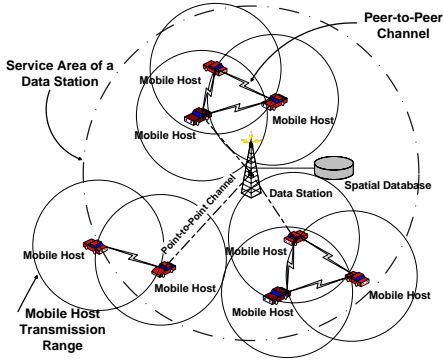


Figure 1. The assumed system environment.

**Cache Management in Mobile Environments** Caching is a key technique to improve data retrieval performance in widely distributed environments. Leveraging the combined resources of several cooperating caches has been proposed to improve file system [4] and Web performance [14]. With the increasing deployment of new P2P wireless communication technologies (e.g., IEEE 802.11b/g and Bluetooth) there exists a new information sharing alternative known as *peer-to-peer cooperative caching* [3]. With this technique mobile hosts communicate with neighboring peers in an *ad-hoc* manner to share information rather than having to rely on the communication link to the remote information sources.

### 3 Problem Statement and Approach

The fundamental idea behind my methodology is to leverage the cached results from prior queries at reachable mobile hosts for answering spatial queries at the local host [10]. To achieve scalability it is imperative that a mobile client can *locally* determine whether the result set from its neighbors provides a full, partial or no answer. As a novel component in our methodology we present verification algorithms that can verify whether a given result object is part of the solution set. We term such an object *verified*. If the object is not guaranteed to be part of the result set, we call it *unverified*.

Figure 1 depicts the operating environment that I am assuming with two main entities: remote spatial databases and wireless mobile hosts. We are considering mobile clients, such as cars, that are instrumented with a global positioning system (GPS) for continuous position information. Furthermore, we assume that two-tiers of wireless connections are available on future automobiles. Traditional, cellular-based networks (such as utilized by the OnStar service) allow medium range connections to base-stations that interface with the wired Internet infrastructure. A second type of short-range networks allow ad-hoc connections with neighboring mobile clients. Technologies that

enable short range 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 lifetime.

### 3.1 Sharing Based Nearest Neighbor Query

With the system infrastructure shown in Figure 1, a mobile host  $q$  can collect NN data from peers to harvest these existing results for completing its own  $k$ NN search. We term this approach a *Sharing Based Nearest Neighbor* (SBNN) query. We propose two approaches to process NN information obtained from peers. The single peer NN verification process, also called  $k$ NN<sub>single</sub>, attempts to verify the validity of  $k$  objects by sequentially verifying results obtained from each single peer. If the number of verified objects is less than  $k$ , then the multiple peer NN verification process,  $k$ NN<sub>multiple</sub>, attempts to complete the verification process with several peers simultaneously.

#### 3.1.1 Step 1: Single Peer NN Verification

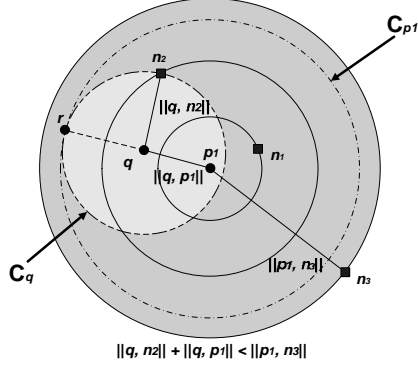
The objective of the  $k$ NN<sub>single</sub> method is to verify whether a point of interest (POI)  $n_i$  obtained from a peer is a valid (i.e., top  $k$ ) nearest neighbor of a mobile host  $q$ . To this end we utilize the spatial relationship between mobile hosts and their POIs as follows.

**Lemma 3.1** *Let  $q$  and  $p_1$  be two mobile hosts, and let  $p_1$  have  $k$  nearest neighbors,  $n_1, n_2, \dots, n_k$ , which are sorted in ascending order according to their distance to  $p_1$ . For any nearest neighbor  $n_i$  of  $p_1$ , if  $\|q, n_i\| + \|q, p_1\| \leq \|p_1, n_k\|$  then  $n_i$  is one of the top  $k$ -nearest neighbors of  $q$ .*

$\|q, n_i\|$ ,  $\|q, p_1\|$ , and  $\|p_1, n_k\|$  are the Euclidean distances between  $q$  and  $n_i$ ,  $q$  and  $p_1$ , and  $p_1$  and its cached farthest nearest neighbor  $n_k$ , respectively (see detailed proof in [9]).

An illustration of Lemma 3.1 is shown in Figure 2. The nearest neighbor  $n_2$  of mobile host  $p_1$ , which is a peer of mobile host  $q$ , can be verified as the nearest neighbor of  $q$  and is termed a *verified nearest neighbor*. Because the Euclidean distance between  $n_2$  and  $q$  plus the Euclidean distance between  $q$  and  $p_1$  is no greater than the Euclidean distance between  $p_1$  and its presently cached farthest nearest neighbor  $n_3$ . The exact ranking of nearest neighbors can also be obtained (see detailed proof in [9]).

The  $k$ NN<sub>single</sub> method maintains a heap  $H$  with the entries of verified and unverified POIs discovered so far. The size of  $H$  is determined by the total number of queried interest objects  $q_k$ . Initially  $H$  is empty and the  $k$ NN<sub>single</sub> method iteratively processes the result set  $P$  of nearest



**Figure 2.** POI  $n_2$  is verified as a valid NN of mobile host  $q$ .

neighbor objects from mobile hosts in the vicinity of  $q$  in sequence. If  $k$  elements in  $H$  are all verified, the  $k$ NN query is fulfilled and  $H$  will remember the top  $k$  NN in sequence. The single peer NN verification procedure is summarized in Algorithm 1.

---

**Algorithm 1**  $k$ NN $_{single}(q, p, k)$

---

```

1:  $H \leftarrow \emptyset$ 
2:  $n_{far} \leftarrow$  the farthest node in  $p.N$ 
3: for  $\forall n_i \in p.N$  and  $|H.verified| < k$  do
4:   if  $\|(q, n_i)\| + \|(q, p)\| \leq \|(p, n_{far})\|$  then
5:      $H.verified \cup = n_i$ 
6:   else
7:     if  $|H| < k$  then
8:        $H.unverified \cup = n_i$ 
9:     else if  $\exists n_j$  and  $\|(q, n_j)\| > \|(q, n_i)\|$  where  $n_j \in H.unverified$  then
10:      replace  $n_j$  in  $H.unverified$  with  $n_i$ 
11:     end if
12:   end if
13: end for
14: return  $H$ 

```

---

### 3.1.2 Step 2: Multiple Peer NN Verification

Under some conditions the  $k$ NN $_{single}$  method may not be able to verify all  $k$  nearest neighbors. Therefore, we extend the verification process to include results from multiple peers simultaneously. The  $k$ NN $_{multiple}$  method combines the *verified region*  $p.R$  of all the peers, each bounded by the outermost NN circle, into a *merged verified region*  $R_v$  by performing the *MapOverlay* algorithm [5] (line 3 of Algorithm 2). The  $k$ NN $_{multiple}$  verification technique is executed based on  $R_v$  similarly to  $k$ NN $_{single}$ . Lemma 3.2 provides the rules for verifying nearest neighbors with multiple peers and the multiple peer NN verification procedure is formalized in Algorithm 2.

**Lemma 3.2** *If the nearest neighbor data set  $P$  is composed of data from  $j$  peers, the merged verified region  $R_v$  can be*

---

**Algorithm 2**  $k$ NN $_{multiple}(q, H, P, k)$

---

```

1:  $R_v \leftarrow \emptyset$ 
2: for  $\forall p \in P$  do
3:    $R_v \cup = p.R$ 
4: end for
5: for  $\forall p \in P$  and  $|H.verified| < k$  do
6:   for  $\forall n_i \in p.N$  and  $n_i \notin H.verified$  do
7:      $C_{n_i} \leftarrow$  create a circle region with  $\|(q, n_i)\|$  as the radius and  $q$  as the center point
8:     if  $C_{n_i} \subset R_v$  and  $|H.verified| < k$  then
9:        $H.verified \cup = n_i$ 
10:    end if
11:   end for
12: end for
13: return  $H$ 

```

---

represented as:

$$R_v = p_1.R \cup p_2.R \cup \dots \cup p_j.R.$$

For any point of interest  $n_i$  in  $R_v$ , the distance between  $q$  and  $n_i$  is used as a radius to create a circle  $C_{n_i}$ . If  $C_{n_i}$  is fully covered by  $R_v$ , then  $n_i$  is a verified NN of  $q$ .

There will be cases when neither  $k$ NN $_{single}$  nor  $k$ NN $_{multiple}$  can fulfill a  $k$ NN query. Hence a set which contains unverified elements is returned. If the response time is critical, a user may agree to accept a  $k$ NN data set with unverified elements, where the objects are not guaranteed to be the top  $k$  nearest neighbors. Otherwise the  $k$ NN query must be forwarded to a spatial database server (Step 3). The partial results in  $H$  can be used to bound and hence speed up the server search process.

### 3.1.3 Step 3: Server $k$ NN Query with Pruning Bounds

We assume that the spatial database server executes an efficient  $k$ -nearest neighbor search algorithm based on R-tree indexing [7] for solving  $k$ NN queries. The NN search is supported with a priority queue containing the nodes visited so far. Initially the priority queue contains the entries of the R-tree root sorted according to their minimum distance (MINDIST) to the query point  $q$ . If the heap  $H$  is full, we can consider the last entry of  $H$  as the final candidate nearest neighbor in the NN search and forward its distance attribute to the server as the branch expanding upper bound. In addition, the distance attribute  $d_v$  of the last verified entry can be another bound, the branch expanding lower bound. Because we are certain about the POIs within the circle region  $C_r$  with radius  $d_v$  and center point  $q$ , the NN search algorithm executed in the server does not need to expand any minimum bounding rectangle which is completely covered by  $C_r$ . To take advantage of the two new bounds for  $k$ NN queries, we slightly modified the  $k$ NN best-first search

---

**Algorithm 3** SBNN( $q, k$ )

---

```
1:  $H \leftarrow \emptyset; P \leftarrow$  peer nodes responding the query request issued
   from  $q$ .
2: for  $\forall p \in P$  and  $|H.verified| < k$  do
3:    $H \cup = kNN_{single}(q, p, k)$  (Step 1)
4: end for
5: if  $|H.verified| < k$  then
6:    $H \cup = kNN_{multiple}(q, H, P, k)$  (Step 2)
7: end if
   {if  $k$  verified NN have been retrieved, or the heap is full and  $q$ 
   accepts unverified results.}
8: if  $(|H.verified| = k)$  or  $(|H| = k$  and  $accept = true)$  then
9:   return  $H$ 
10: end if
   {if  $H$  is not full or  $q$  denies any unverified results, forward the
   query with the branch expanding upper and lower bounds to
   the database server.}
11:  $H \leftarrow kNN$  query results returned from the server. (Step 3)
12: return  $H$ 
```

---

algorithm such that it calculates one more metric, the maximum distance (MAXDIST), for pruning R-tree branches. MAXDIST indicates which MBRs are totally covered in region  $C_r$  and the algorithm does not need to expand them. The complete procedure of the sharing based  $kNN$  query is described in Algorithm 3.

### 3.2 Sharing Based Window Query

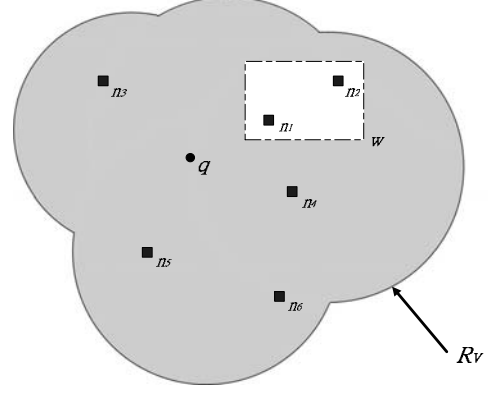
For a *Sharing Based Window Query* (SBWQ), a mobile host  $q$  has to merge verified regions ( $p.R$ ) and collect related POI data from peers. Then  $q$  computes the spatial relationship between the query window  $w$  and the merged verified region  $R_v$ . If  $w$  can be totally covered by  $R_v$ , the window query can be fulfilled. Otherwise, the whole or part of the query window must be forwarded to a spatial database server.

#### 3.2.1 Step 1: Verified Region Merging

The SBWQ algorithm first combines the verified regions  $p.R$  of all the peers, each bounded by the outermost NN circle, into a merged verified region  $R_v$ . Then it calculates the spatial relationship between the query window  $w$  and  $R_v$ . If  $w$  entirely falls inside  $R_v$ , SBWQ will return the POIs which overlap with  $w$  (Figure 3).

#### 3.2.2 Step 2: Server Window Query with Partial Query Window

There will be cases when the SBWQ algorithm can provide only a partial or no result to a window query. Consequently an updated (reduced) query window  $w'$  will be forwarded to the spatial database server for speeding up the server search process. The SBWQ algorithm is formalized in Algorithm 4.



**Figure 3.** POI  $n_1$  and  $n_2$  are the query results of this sharing based window query.

---

**Algorithm 4** SBWQ( $q, w$ )

---

```
1:  $P \leftarrow$  peer nodes responding the query request issued from  $q$ .
2: for  $\forall p \in P$  do
3:    $R_v \cup = p.R$  (Step 1)
4: end for
5: if  $w \subset R_v$  then
6:    $RQ \leftarrow$  POIs which are overlapped with  $w$ 
7: end if
   {if  $w \not\subset R_v$ , forward the updated  $w$  to the database server.}
8:  $RQ \leftarrow$  query results returned from the server. (Step 2)
9: return  $RQ$ 
```

---

## 4 Extension to Wireless Broadcast Environments

From a high level perspective, there are two approaches for mobile data access. One is the *on-demand access* model and the other is the *wireless broadcast model*. For the on-demand access model, servers process queries submitted from mobile clients and return the results with point-to-point connections. For the wireless broadcast model, servers continuously broadcast information in wireless channels and clients are responsible for receiving their desirable data. According to previous research [8, 16], the major advantage of the wireless broadcast model compared with the on-demand model is that it allows simultaneous access by an arbitrary number of mobile users and it can be a suitable data dissemination mechanism for solving spatial queries in mobile environments. However if a mobile host missed its requested data packets, it has to wait for the next broadcast cycle and this is the main limitation of the broadcast model. One extension of my approach is to combine the P2P sharing mechanism with the wireless broadcast model. For most spatial queries, mobile hosts can retrieve data objects from peers for fulfilling their queries rather than waiting for data packets from the broadcast channel. Consequently, we can significantly decrease the access latency but still keep excellent scalability [11].

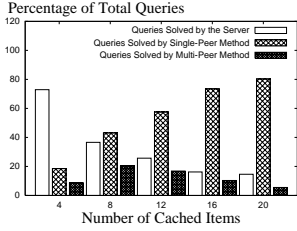


Fig. 4a. Sharing based  $k$ NN queries.

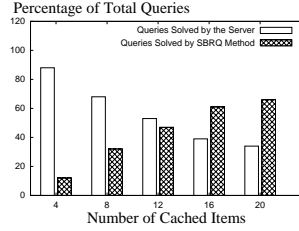


Fig. 4b. Sharing based range queries.

**Figure 4. The percentage of queries that are resolved by peers and the server as a function of the mobile host cache capacity.**

## 5 Experimental Validation

To evaluate the performance of our approach we have implemented the sharing based spatial query algorithms within a simulator. In addition to enabling robust and decentralized applications, the objective of our peer-to-peer design is to increase scalability in two dimensions. First, the server access workload can be reduced as queries are answered directly by peers. Second, for the remaining queries that must be sent to the server, our technique diminishes the server search overhead by simplifying the spatial queries. Consequently, the focus of our simulations is on quantifying the server load variations. We have performed our experiments with both synthetic and real-world parameter sets (see detailed simulation results in [9]).

Figure 4 illustrates cache capacities from 4 to 20 with the real-world data set. We conclude from all the performed experiments that the mobile host density has a considerable impact on the server access rate. As a result, if more mobile hosts travel in a specific area, each MH has a higher opportunity to fulfill its spatial queries by peers. Furthermore, for those queries that must be forwarded to the server, our algorithms successfully reduce processing for them and decrease the server load.

## 6 Conclusions

I have presented a novel approach for answering spatial queries by leveraging results from neighboring peers within a mobile environment. Significantly, our method allows a mobile peer to locally verify whether candidate objects received from neighbors are indeed part of its own query result data set. The simulation results indicate that the technique can reduce the server access traffic by a significant amount, for example up to 80% in a dense urban area. 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 load on the remote databases increases sub-linearly with the number 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] G. Cao, L. Yin, and C. R. Das. Cooperative Cache-Based Data Access in Ad Hoc Networks. *IEEE Computer*, 37(2):32–39, 2004.
- [3] 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.
- [4] M. Dahlin, R. Y. Wang, T. E. Anderson, and D. A. Patterson. Cooperative caching: Using remote client memory to improve file system performance. In *Proceedings of the 1<sup>st</sup> USENIX OSDI*, pages 267–280, November 1994.
- [5] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications (2nd Edition)*. Springer, 2000.
- [6] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 47–57, Boston, Massachusetts, June 18–21, 1984.
- [7] G. R. Hjaltason and H. Samet. Distance browsing in spatial databases. *ACM Trans. Database Syst.*, 24(2):265–318, 1999.
- [8] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Data on air: Organization and access. *IEEE Trans. Knowl. Data Eng.*, 9(3):353–372, 1997.
- [9] W.-S. Ku, R. Zimmermann, and C.-N. Wan. Location-based Spatial Queries with Data Sharing in Mobile Environments. Technical Report USC-CS-TR05-843, University of Southern California, 2005.
- [10] W.-S. Ku, R. Zimmermann, C.-N. Wan, and H. Wang. Maple: A Mobile Scalable P2P Nearest Neighbor Query System for Location-based Services. In *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, Atlanta, USA, 2006*.
- [11] W.-S. Ku, R. Zimmermann, and H. Wang. Location-based Spatial Queries with Data Sharing in Wireless Broadcast Environments. *Submitted for publication*, 2006.
- [12] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *SIGMOD Conference*, pages 71–79, 1995.
- [13] T. K. Sellis, N. Roussopoulos, and C. Faloutsos. The R+-Tree: A Dynamic Index for Multi-Dimensional Objects. In *VLDB*, pages 507–518, 1987.
- [14] D. Wessels and K. Claffy. ICP and the Squid Web Cache. *IEEE Journal on Selected Areas in Communications (JSAC)*, pages 345–357, March 1998.
- [15] J. Zhao and G. Cao. Vadd: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. In *INFOCOM, Barcelona, Spain, 2006*.
- [16] B. Zheng, W.-C. Lee, and D. L. Lee. Spatial queries in wireless broadcast systems. *Wireless Networks*, 10(6):723–736, 2004.