**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

*IJCSMC, Vol. 3, Issue. 7, July 2014, pg.1 – 5*

**SURVEY ARTICLE**

# Survey of Creating Location Based Regions of Spatial Queries Using Proxy Approach in Mobile Environments

**[1]K.Suresh Babu, [2]Swetha Madireddy**

[1]Assistant Professor M.Tech (CS), School Of Information Technology, JNTUH, India

[2]M.Tech in Department of CNIS at School Of Information Technology, JNTUH, India

[1] kare_suresh@yahoo.com

[2] swetha.sunny20@gmail.com

***Abstract:** Caching valid regions of spatial queries at mobile clients is effective in reducing the number of queries submitted by mobile clients and query load on the server. However, mobile clients suffer from longer waiting time for the server to compute valid regions. We propose in this paper a proxy-based approach to continuous nearest-neighbor (NN) and window queries. The proxy creates estimated valid regions (EVRs) for mobile clients by exploiting spatial and temporal locality of spatial queries. For NN queries, we devise two new algorithms to accelerate EVR growth, leading the proxy to build effective EVRs even when the cache size is small. On the other hand, we propose to represent the EVRs of window queries in the form of vectors, called estimated window vectors (EWVs), to achieve larger estimated valid regions. This novel representation and the associated creation algorithm result in more effective EVRs of window queries. In addition, due to the distinct characteristics, we use separate index structures, namely EVR-tree and gridindex, for NN queries and window queries, respectively. To further increase efficiency, we develop algorithms to exploit the results of NN queries to aid grid index growth, benefiting EWV creation of window queries. Similarly, the grid index is utilized to support NN query answering and EVR updating. We conduct several experiments for performance evaluation. The experimental results show that the proposed approach significantly outperforms the existing proxy-based approaches.*

***Keywords:** Nearest neighbor query, window query, spatial query processing, location-based service, mobile computing*

 

## I. Introduction

Spatial queries are one of the most important LBSs. According to spatial constraints, spatial queries can be divided into several categories including nearest neighbor (NN) queries and window queries. An NN query is to find the nearest data object with respect to the location at which the query is issued. For example, a user may launch an NN query like "show the nearest coffee shop with respect to my current location" On the other hand, a window query is to find the object within a specific window frame. An example window query is "show all restaurants in my car navigation window. In general, a mobile client continuously launches spatial queries until the client obtains a satisfactory answer. For Example, a query "show me the rate of the nearest hotel with respect to my current location" is continuously submitted in a moving car so as to find a desired hotel. The naive method answering continuous spatial queries is to submit a new query whenever the query location changes. Fortunately, in the real world, the queries of a continuous query usually exhibit spatial locality. Thus, caching the query result and the corresponding valid region (VR) in the client-side cache to mitigate the above problems. The valid region, also known as the valid scope, of a query is the region where the answer of the query remains valid. Subsequent queries can be avoided as long as the client is in the valid region. Note that a VR is associated with a query by definition. However, by associating a VR with the corresponding object, the VR of an object p can be used to resolve all the queries whose answer object is p. Although VRs are useful, an LBS server may not provide VRs in practice since the server may simply provide query results only or would not compute VRs under heavy load. In these situations, mobile service providers (e.g., Verizon Wireless and AT&T) or smart phone makers (e.g., Apple and RIM) can utilize a proxy architecture where the proxy provides estimated valid regions (EVRs), which are the sub regions of the corresponding VRs, for the clients. With the proxy architecture, the clients still can enjoy the benefits of EVRs even if the LBS server does not provide VRs and thus the mobile service providers and smart phone makers can attract more clients. In other words, the proxy-based approach is an alternative solution to serve the function of VRs in case that the LBS server could not provide VRs. The authors proposed to deploy a proxy between mobile clients and the LBS server to build EVRs by exploiting the queries interested in the same objects. EVR provisioning is inspired by that, in addition to spatial locality, spatial queries also exhibit temporal locality, resulting from that a number of queries with close query locations are likely to be launched during a short interval. For instance, a large number of NN queries about nearest hotels will be launched by passengers after a train arrives. Hence, a proxy may be able to answer subsequent queries interested in the same objects by caching EVRs created based on previous queries. For NN queries, we devise new algorithms to efficiently create new and extend existing EVRs. The devised algorithms not only enable mobile clients to obtain effective EVRs immediately but also lead the proxy to build effective EVRs even when the proxy cache size is small. For window queries, we propose to index the positions of data objects, instead of EVRs, by a grid index. Since VRs of window queries may not be convex polygons, creating effective polygonal EVRs by previous queries is inherently difficult. Thus, we propose to represent the EVRs of window queries in the form of vectors, called estimated window vectors (EWVs), to achieve larger estimated valid regions. Such novel representation and indexing lead the proxy to efficiently create more effective EVRs of window queries. We introduce two index structures, an EVR-tree for NN queries and a grid index for window queries, due to the distinct characteristics of NN and window queries. We also develop algorithms to make these two index structures mutually support each other. Specifically, the grid index can be used to answer NN queries and extend existing EVRs. The answer objects of NN queries are exploited to update the grid index, benefiting the creation of more effective EWVs of window queries. We conduct several experiments to compare the proposed approach with the existing proxy based approaches and the representative server based approach. The experimental results show that the proposed approach significantly outperforms the existing proxy-based approaches. Moreover, we compare the proposed approach with the representative server-based approach for understanding the benefits of the proxy architecture. The experimental results demonstrate that the performance of the proposed approach is close to that of even though the proposed approach has only partial information of data objects.

 

## II. Related Work

In recent years, a significant number of research studies have been proposed for spatial query processing. Most of these studies addressed representative spatial queries, such as NN queries, kNN queries, and window queries. For different scenarios, some studies coped with static data objects while some tackled mobile data objects. Since our work falls into the former category, we focus on reviewing previous studies on static data objects in the following. R-tree and its variants, such as R_-tree, are one of the most popular methods of static spatial query processing. An LBS server is able to answer spatial queries quickly using Rtree- like index structures. However, in mobile environments, mobile clients usually launch a continuous query consisting of a number of queries with different query locations for obtaining a satisfactory answer. Continuous queries cause the conventional scheme to suffer from wireless medium contention and heavy query load. To address this problem, previous studies proposed that mobile clients could avoid launching unnecessary queries by caching VRs and EVRs. According to the architecture, these studies can be classified into two categories: server-based approaches and proxy based approaches. Basically, server-based approaches have the complete information of data objects and can utilize the information to create VRs for mobile clients. On the other hand, proxy based approaches have only partial information of objects and exploit spatial and temporal locality of queries of mobile clients to build EVRs. We first introduce the existing server-based approaches as follows: proposed that an LBS server creates the Voronoi diagram of data objects in an offline manner and returns the answer object of an NN query as well as the corresponding Voronoi cell to the querying client. The client caches the Voronoi cell to reduce the number of subsequent queries since the Voronoi cell is the VR of the answer object. The main advantage is that the LBS server constructs the Voronoi diagram once and uses it repeatedly. However, searching the corresponding Voronoi cell is time consuming and object updates incur the overhead of partial reconstruction of the Voronoi diagram. Zheng et al. introduced a grid-partition-index to improve search efficiency, but it is still impaired by object updates.

Proxy Design:

A proxy builds EVRs of NN queries and EWVs of window queries based on NN query history and available data objects, respectively. The proxy maintains an object cache and two index structures: an EVR-tree for NN queries and a grid index for window queries. The two index structures share the data objects in the object cache. The EVR-tree is an R-tree (or its variants) composed of EVRs where each EVR is wrapped in a minimum bounding box (MBR). An EVR consists of the region vertices with respect to a data object and a pointer to the corresponding object entry in the object cache. When an NN query point q is located in an EVR of the EVR-tree, the proxy retrieves the corresponding object from the object cache to answer the query. On the other hand, the service area is divided into m _ n grid cells managed by the grid index. Grid cells are classified into two categories: fully cached cells and uncached cells. All grid cells are initialized to uncached. The proxy marks a cell as fully cached when all the objects within the cell are received. The corresponding grid index entry of a fully cached cell caches the object pointers to the associated object entries in the object cache. The purpose of fully cached and uncached cells is to realize the stored object distribution, enabling the proxy to create EWVs of window queries effectively. When receiving a window query, the proxy obtains the result and creates the corresponding EWV by retrieving stored objects in the surrounding fully cached cells. Although the EVR tree and the grid index are designed for NN and window queries, respectively, these two index structures mutually support each other.

System Architecture**:**

Depicts the proposed system architecture for NN and window query processing. The system architecture consists of three parts: 1) an external LBS server, 2) deployed proxies, and 3) the mobile clients. The LBS server is responsible for managing static data objects and answering the queries submitted by the proxies. Note that the LBS server can use any index structure (e.g., R-tree or grid index) to process spatial queries. The LBS server is assumed not to provide VRs. Each of the deployed proxies supervises one service area and provides EVRs of NN queries and EWVs (vector form of EVRs) of window queries for mobile clients in the service area. Each base station serves as an intermediate relay for queries and query results between mobile clients and the associated proxy. Base stations, proxies, and the LBS server are connected by a wired network. A mobile client maintains a cache to store the query results and the corresponding

 

EVRs. When a mobile client has a spatial query, the mobile device first examines whether the current location is in the EVR of the stored result. If so, the stored result remains valid and the mobile device directly shows it to the client. Otherwise, the mobile device submits the query, which is received and then forwarded by the base station, to the proxy. For the received query, the proxy will return the query result as well as the corresponding EVR to the client.

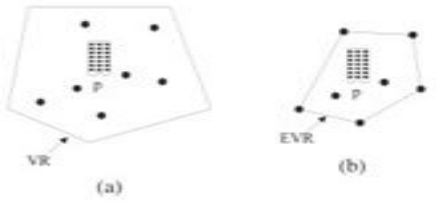


**Fig1**: Example VR and EVR. (a) An example VR. (b) An example EVR.

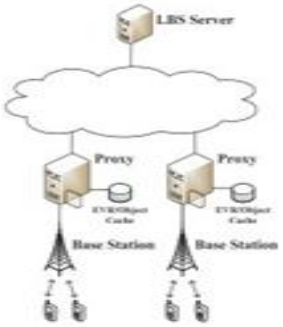


**Fig2:** System architecture.

Nearest-Neighbor Query Processing:

When receiving an NN query, a proxy first attempts to answer the query with the EVR-tree and the grid index. If the proxy cannot answer the query, it will submit one or two 2NN queries to the LBS server. The rationale of extending the received NN query into 2NN queries is that the distance between the nearest and second nearest objects is useful for building the EVR of the nearest object based on the theoretical result by exploiting the objects returned by the LBS server, the proxy extends an existing or creates a new EVR. Besides, the returned objects are utilized to aid grid index growth, which will facilitate window query processing. For each NN query, the proxy provides the mobile client not only the answer object (the nearest object) but also an EVR for helping the client avoid subsequent queries. Note that these 2NN queries initially cause slightly heavier load on the LBS server, but they lead the proxy to provide effective EVRs for mobile clients efficiently. With the EVRs, the number of queries submitted by the clients and the load on the LBS server are reduced greatly.

## III. CONCLUSION

A proxy-based approach to continuous NN and window queries in mobile environments. The proxy takes advantage of spatial and temporal locality of spatial queries to create EVRs of NN and window queries. Different from prior work, we devised new EVR creation and extension algorithms for NN queries, enabling the proxy to build effective EVRs efficiently. The devised algorithms make the proxy achieve high performance even when the cache size is small. On the other hand, we propose to represent EVRs of window queries in the form of vectors called estimated window vectors, to achieve larger estimated valid regions. Moreover, due to distinct characteristics, we introduce an EVR-tree and a grid index to process NN and window queries, respectively. The algorithms for mutual support of the EVR-tree and the grid index are developed to further enhance the system performance. The experimental results show that the proposed approach significantly outperforms the existing proxy-based approaches since our proposed algorithms create much larger EVRs for mobile clients.

 

## References:

[1]D. Lee, B. Zheng, and W.-C. Lee, "Data Management in Location- Dependent Information Services," IEEE Pervasive Computing, vol. 1, no. 3, pp. 65-72, July-Sept. 2002.
[2] B. Zheng, J. Xu, and D.L. Lee, "Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments," IEEE Trans. Computers, vol. 15, no. 10, pp. 1141- 1153, Oct. 2002.
[3] B. Zheng and D.L. Lee, "Processing Location-Dependent Queries in a Multi-Cell Wireless Environment," Proc. Second ACM Int'l Workshop Data Eng. for Wireless and Mobile Access, 2001.
[4] B. Zheng, J. Xu, W.-C. Lee, and D.L. Lee, "On Semantic Caching and Query Scheduling for Mobile Nearest-Neighbor Search," Wireless Networks, vol. 10, no. 6, pp. 653-664, Dec. 2004.
[5] X. Gao and A. Hurson, "Location Dependent Query Proxy," Proc. ACM Int'l Symp. Applied Computing, pp. 1120-1124, 2005.
[6] X. Gao, J. Sustersic, and A.R. Hurson, "Window Query Processing with Proxy Cache," Proc. Seventh IEEE Int'l Conf. Mobile Data Management, 2006.

## Author Profiles:

**Swetha Madireddy,** Pursuing M.Tech in Department of CNIS at School Of Information Technology JNTUH, Kukatpally, Hyderabad Rangareddy dist., A.P., India. Her research interest includes in Security, Networks.
**K.Suresh Babu,** Assistant Professor M.Tech(CS), School Of Information Technology JNTUH, Kukatpally, Hyderabad, Rangareddy dist., A.P., India.

 