Journals: International Journal of Software & Hardware Research in Engineering
ISSN-2347-4890
Volume 4 Issue 2 February, 2016

A Balance Method For Continuous Location Based Spatial Queries for Mobile Environments.

Miss. Saili Khalde[1]; Prof. Priyanka More [2]
ME Student,GSMCOE[1]; Research Scholar, GSMCOE [2]
sailikhalde@gmail.com[1]; priyankamored@gmail.com[2]

ABSTRACT

In location-based environments, a large number of servers and a large number of continuous spatial queries are effectively reduced to catching valid region in obile clients; however, mobile clients suffer from longer waiting time of the server to compute. We propose a novel proxy-based approach to continuous nearest-neighbor (NN) and having window queries. The approximate valid regions (EVRs) created by proxy for mobile user by exploiting spatial and it temporal locality of spatial queries; to accelerate EVR's growth we implement two new algorithms , leading the proxy to build effective EVRs even when the cache size is small. For above solutions, we propose to represent the EVRs for the window queries in the form of vectors, called estimated window vectors (EWVs), which provide larger roughly calculated valid regions. This entire representation with above two algorithms leads too effective EVRs of window queries. Due to the addition of distinct characteristics, we use the separate index structures, namely grid index and EVR-tree for window queries and NN queries, respectively. We gain accuracy in addition by applying efficient load balancing technique and use of multiple backup servers. For handling of frequent objects updates we will introduce the recent cost-effective method in order to overcome limitations of existing approach.

Keywords: Nearest neighbor query; window query; spatial query processing ;location-based service; mobile computing.

1. INTRODUCTION

The integration of position locators and mobile devices enables new location-aware environments where all objects of interest can determine their locations. In such environments, moving objects are continuously changing locations and the location information is sent periodically to server .Spatial queries are one of the key principles of lbss. As per spatial constraints, spatial queries can be split into several categories including nearest neighbor (NN) queries and window queries. An NN query is to detect the nearest data object with respect to the location at which the query is issued. Consider, for instance, a user (mobile client) in a non-familiar city, who would like to know the 10 nearest restaurants. This is an instance of a k nearest neighbor (knn) query, where the client current location is query point of the client and the set of data objects contains the city restaurants. As another solution is, the user can ask for each restaurants located within a certain distance, i.e., in 200 meters. This is a range query instance.

In general, a mobile client continuously launches spatial queries until the client obtains an acceptable 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. Answering continuous spatial queries is natural method to submit a new query whenever the query location changes. The natural method is able to provide correct results, but following are the problems faces:

High power consumption.: The power consumption of a mobile device is high-level since the mobile device keeps submitting queries to the LBS server.

Heavy server load.: A steady query usually consists of a number of queries to the LBS server, thereby increasing the load on the LBS server.

We represented new approach in which proxy

architecture as well as several companion algorithms is presented to provide evrs of NN and window queries on static data objects for mobile clients. This method is essentially showing good efficiency and security against the existing methods. However we need this method to improve for reliability as well as efficient method required for data object updates management.

2. LITERATURE REVIEW

Rtree like index structures. Though, in mobile environments, mobile previously in literature survey we are going to discuss all recent methods over the Continuous Location-Based Spatial Queries for Mobile Environments.

In recent years, a significant number of research studies have been proposed for spatial query processing. Most of these studies show spatial queries, such as knn queries, window queries, and NN queries. For alternate scenarios, some studies coped with static data objects while some tackled mobile data objects. Further studies coped with continuous window query monitoring while addressed continuous knn query monitoring. Since our work sink into the old category, we concentrate on reviewing previous studies on static data objects in the following.

R-tree and its variants, such as R-tree [1], are one of the most popular methods of static spatial query processing. An LBS server is handling answer spatial queries quickly using clients usually launch a continuous query consisting of a number of queries with different query locations for obtaining a sufficient answer. Continuous queries cause the conventional scheme to suffer from wireless medium contention and massive query load. To address this problem, previous studies proposed that mobile clients could avoid launching unnecessary queries by caching vrs and evrs .In accordance with the architecture, these studies can be organize 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. For further, 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: In [2], Zheng et al. Proposed that an LBS server creates the Voronoi diagram [18] 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. [19] introduced a grid-partition index to improve search efficiency, but it is still impaired by object updates.

In B. Zheng and D.L. Lee "[3] Processing Location-Dependent Queries in a Multi-Cell Wireless Environment". This paper is based on common scenario where data objects are stationary while clients, which issues queries, are mobiles. For query processing they use Voronoi Diagrams to construct index and semantic cache for improving data reusability. For handoff clients; this paper proposed three scheduling methods namely the priority method, the intelligent method and the hybrid method to improve performance.

B. Gedik and L. Liu,"[4] Mobieyes: A Distributed Location Monitoring Service Using Moving Location Queries "An important research challenge for modern location-based services is the scalable processing of location monitoring requests on a large collection of mobile objects. The centralized architecture, though studied extensively in literature, would create intolerable performance problems as the number of mobile objects grows significantly. This paper presents a distributed architecture and a suite of optimization techniques for scalable processing of continuously moving location queries. Moving location queries can be viewed as standing location tracking requests that continuously monitor the locations of mobile objects of interest and return a subset of mobile objects when certain conditions are met. This describes the design of mobieyes, a distributed real time location monitoring system in a mobile environment. The main idea behind the mobieyes' distributed architecture is to promote a careful partition of a real time location monitoring task into an optimal coordination of server-side processing and client-side processing. Such a partition allows evaluating moving location queries

with a high degree of precision using a small number of location updates, thus providing highly scalable location monitoring services. A set of optimization techniques are used to limit the amount of computation to be handled by the mobile objects and enhance the overall performance and system utilization of mobieyes. Important metrics to validate the proposed architecture and optimizations include messaging cost, server load, and amount of computation at individual mobile objects. We evaluate the scalability of the mobieyes location monitoring approach using a simulation model based on a mobile setup. This paper experimental results show that mobieyes can lead to significant savings in terms of server load and messaging cost when compared to solutions relying on central processing of location information.

H. Hu, J. Xu, and D.L. Lee"[5] A Generic Framework for Monitoring Continuous Spatial Queries over Moving Objects" This paper proposes a generic framework for monitoring continuous spatial queries over moving objects. The frame-work distinguishes itself from existing work by being the first to address the location update issue and to provide a common interface for monitoring mixed types of queries. Based on the notion of safe region, the client location update strategy is developed based on the queries being monitored .Thus, it significantly reduces the wireless communication and query reevaluation costs required to maintain the upto-date query results. This paper proposes algorithms for query evaluation/reevaluation and for safe region computation in this framework. Enhancements are also proposed to take advantage of two practical mobility assumptions: maximum speed and steady movement. The experimental results of this paper show that proposed framework substantially outperforms the traditional periodic monitoring scheme in terms of monitoring accuracy and CPU time while achieving a close-to-optimal wireless communication cost. The framework also can scale up to a large monitoring system and is robust under various object mobility patterns.

3. PROPOSED SYSTEM

3.1 System Architecture:

Fig. 1 depicts the proposed system architecture. The goal of presenting this approach is to overcome the problems associated with existing methods such as Lack of reliability and load balancing, inefficient method for handling frequent objects updates, slow EVR growth, and lack of mutual support. The slow EVR growth and lack mutual support issues are overcome by our investigated method. In this project same method further modified for the better reliability as well as efficiently handling of frequent objects updates. Reliability will achieve by efficient load balancing and use of multiple backup LSB servers. For the efficient handling of frequent objects updates we will introduce the recent method in order to overcome limitations of existing approach. Practical work will carry using Java and J2ME.

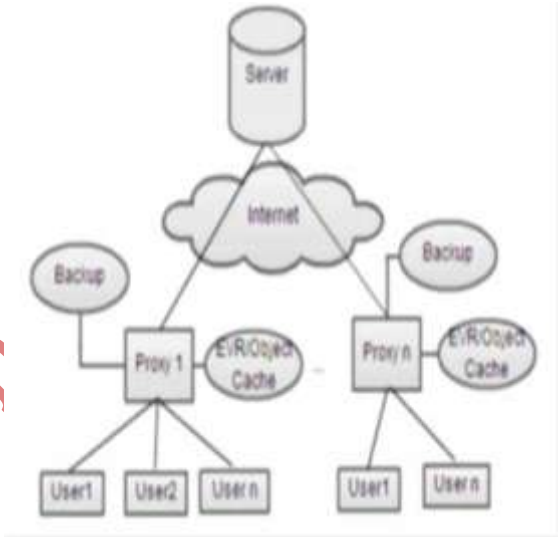

Fig. 1: Proposed Architecture.

3.2 Load Balancing concept:

In load balancing we consider 3 scenarios 1) When any proxy server having more request than threshold value then there is chance to collapse the server, which may causes data loss. To avoid this problem we give back facility to each proxy. When proxy get collapse all live transactions are saved to backup and when proxy comes to working state then proxy will recover from back up data. 2) While transaction if proxy fails due to any physical reasons then there is chances of data loss. To overcome this problem we introduce method "load forwarding". In this case live transactions are passing over other proxy to avoid data loss. 3) For each proxy it has its capacity. If user request of proxy exceeds limit of its capacity then its performance get degraded. So in this case we forward the proxy overload to another proxy who has lower load than its capacity.

3.3 Algorithm Used:

Input: spatial query

Process:

Step 1: design proxy server.

Step 2: resolve query using nearest neighbor.

Step 3: create the estimated valid regions.

Step 4: create the estimated window vectors.

Step 5: apply load balancing mechanism.

Output: result of query.

3.4 Mathematical Model:

System can be Describe through mathematically. For mathematical model we consider S will describe total system.

So S will be,

S = {Input, Output, Process}

Detail of each element is given bellow,

S = System

VR = Valid Regions

NN = Nearest Neighbor

EVR = Estimated Valid Regions

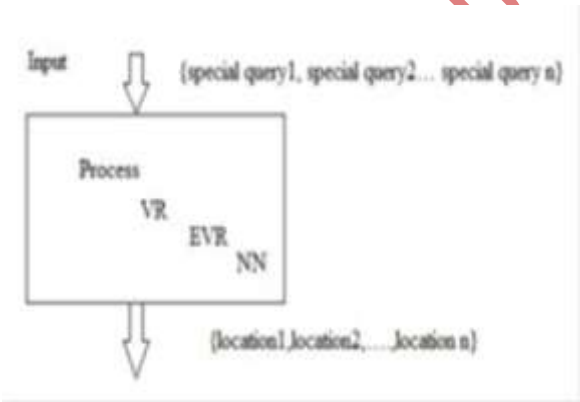

Fig. 2: Location-Based Spatial Queries

Input:

We input special query set as input so input set will be {Special query1, special query2… special query n}

Output:

As we pass special query as input data set so as output its will give special location to user.

{location1, location2… location n}

Process:

For getting appropriate location from given set of special query, query has to go through number of processes that will be described below.

1. Proxy Design:

ID-Identity

e_{ID} - denotes the corresponding EVR ID

Cell flag- flag indicates whether p is located in a fully cached cell Object info- contains the basic information of p

P - is located in a fully cached cell

$< ID, (x,y), e_{ID}, cellflag, object\_info >$

An entry of the object cache is in the form of

$< ID, (x,y), e_{ID}, cellflag, object\_info >.$

Each data object p is assumed to have a unique ID. (x, y) Is the 2D coordinating of p? e_{ID} Denotes the corresponding $EVRID$ and cell flag indicates whether p is located in a fully cached cell. Objectinfo contains the basic information of p.

2. Nearest-Neighbor Query Processing:

3. Resolving NN Queries by Grid Index

C - Circle

$Dist$ - Distance

p – Point p

q – Point q

$dist(q, p_2)$

$dist\ (p; q)$

When the proxy cannot answer an NN query by the EVR tree, it attempts to exploit the grid index and cached objects to resolve the query. With the grid index, the proxy examines whether the query point q lies in a fully cached cell. If so, the proxy

retrieves the nearest object p_1 and the second nearest object p_2 to q via the grid index. Besides p_1, the proxy retrieves p_2 since we intend to update $EVR(P_1)$ in case that $EVR(P_1)$ exists. With p_1 and p_2, the proxy calculates $dist\,(q;\, p_2)$ and creates a circle C centered at q with $dist\,(q;\, p_2)$ as the radius. Note that $dist\,(p;\, q)$ denotes the Euclidean distance between any two given point's p and q. If the circle C does not overlap any UN cached cell, p_1 and p_2 can be guaranteed to be the real nearest and second nearest objects to q,

4. EVR Creation

When neither the EVR-tree nor the grid index can be used to answer the NN query with the location (xq, yq), the proxy will create a new EVR for the answer object p1 of the query

$$Dist\,((q;\, p_1) < dist\,(q;\, p_1))$$

$$r_1 = (dist\,(q;\, p_2) - dist\,(q;\, p_1)) = 2. \quad (1)$$

$$r_2 = dist\,(p_1;\, p_3) = 2. \quad (2)$$

Next, the proxy submits another 2NN query with the query location $(x_1;\, y_1)$ to the LBS server. Obviously, the nearest object to the location $(x_1;\, y_1)$ is p_1. Let the second nearest object of this new 2NN query be p_3 with the location (x_3,y_3). Similarly, we build another circle C_2 with center $(x_1;\, y_1)$ and radius r_2

, where $r_2 = dist\,(p_1;\, p_3) = 2$.

$vnew_1$ ($xnew_1$, $ynew_1$),

$vnew_2$ ($xnew_2$, $ynew_2$),

$vnew_3$ ($xnew_3$, $ynew_3$),

$vnew_4$ ($xnew_4$, $ynew_{42}$),

$vnew_5$ ($xnew_5$, $ynew_5$),

$vnew_6$ ($xnew_6$, $ynew_6$),

5. Updating Grid Index by Results of NN Queries

6. EVR of knn Query

7. EWV Creation

The polygonal $EVRs$ are extremely small and in effective. To overcome this problem, we propose to cache data objects indexed by the grid index and represent the $EVRs$ in the form of vectors, called estimated window vectors, to enlarge the estimated valid regions. Due to rectangular windows and cached objects, the $EWVs$ can be set effectively. An EWV consists of estimated valid distance components and is denoted by

$$V =< Dxne, Dyne, Dxnw, Dynw, Dxse, Dyse, Dxsw, Dysw > (3)$$

Each component is defined as the estimated valid distance with respect to a specific direction. For example, $Dxne$ and $Dyne$ are defined as the east and north estimated valid distances with respect to the northeast direction, respectively.

For EWV creation, two requirements must be met:

1) All answer objects p_0 is must remain valid and

2) No outer objects o_0 is will be encountered. The data objects outside the query window are called outer objects. Since the value of each component is determined in a similar manner, we use $Dxne$ and $Dyne$ for the northeast direction to explain the process. To simplify the process of determining $Dxne$ and $Dyne$, the proxy first calculates the estimated valid distances De and Dn for the query window moving east and north, respectively. Determining De and Dn in advance helps the proxy calculate $Dxne$ and $Dyne$ because the query window heading northeast may encounter the o_0 is in the east and north directions .For De ,the proxy first calculates the minimum distance between p_0 is and the left side LW of the query window to ensure the validity of p_0 is. The minimum distance $D\,ie$ is set to,

$$Die = min\{(dist(p_0, LW)\}: \quad (4)$$

In this section we are presenting the current state of implementation and results achieved.

4. EXPERIMENTAL ANALYSIS

Results of Practical Work:-

In section, we evaluate the proposed system and compare its performance with the existing system. The below fig clearly shows that communication efficiency of proposed system is better than existing system. As number of nodes is added the communication cost of existing system also increases, while compare with proposed system the communication cost is slightly increases as number of node increases.

Graph of Communication Cost

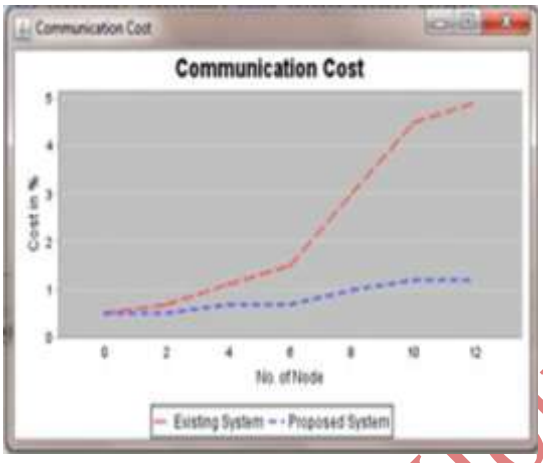

Fig. 3: Communication cost between proposed and existing system

Graph of Load Balance

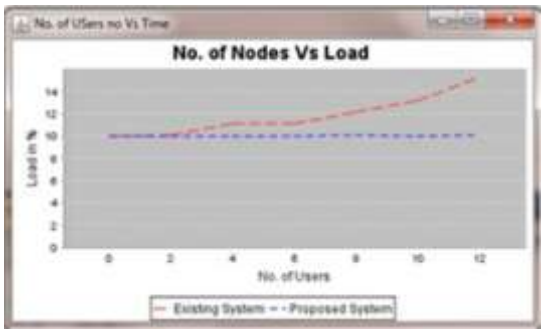

Fig. 4 :Load balance between proposed and existing system

In Figure 4 we can see that the effect load between existing system and proposed system. As the fig. Shows load balance ratio, existing system load increases as the number of nodes are increases, while in proposed system the load is constant as number of user increases.

Graph of Communication Cost

For NN queries, the increase in the number of mobile client's results in more submitted, causing the performance of existing and proposed approaches to deteriorate in terms of query answering time and server load queries. Fig5 shows the ratio of query answering time of existing system and proposed system. Existing system require more time to answer particular query of node than time taken by proposed system

Fig. 5: Performance of Query answering time versus number of clients

5. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel algorithm for a proxy-based approach for NN continuous and window queries in mobile environments. The proxy takes extra benefit of spatial and temporal locality of spatial queries to create evrs of NN and window queries different from previous work, Another EVR creation and extension algorithms for NN queries, grant the proxy to build cost-effective evrs efficiently. The originate algorithms make the proxy achieve high performance even when the cache size is small. For more productiveness, we develop algorithms to exploit the results of NN queries to grid index growth, be advantageous to EWV creation of window queries. Comparably, the grid index can be utilized to support the NN query answering or EVR updating. We have executed several experiments for the performance evaluation. The experimented results clears proposed view significantly outperforms existing proxy-based approaches. Cost-Effectiveness will achieve by efficient load balancing and use of multiple back LSB servers. To efficiently update frequent objects we will introduce the recent method in order to overcome limitations of existing approach. Practical work will carried out using

Java and J2ME.

6. ACKNOWLEDGMENT

The specially thank to my guide "Prof. Priyanka More" and department of computer Engg. Of GSM COE, Balewadi giving her contribution in writing paper.

7. REFERENCES

[1]. YN. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger-"The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 322-331, 1990.

[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 Queriesin a Multi-Cell Wireless Environment," Proc. Second ACM Int'l Workshop Data Eng. for Wireless and Mobile Access, 2001.

[4]. B. Gedik and L. Liu, "Mobieyes: A Distributed Location Monitoring Service Using Moving Location Queries," IEEE Trans. Mobile Computing, vol. 5, no. 6, pp. 1384-1042, Oct. 2006.

[5]. X. Gao, J. Sustersic, and A.R. Hurson, "Window Query Processingwith Proxy Cache," Proc. Seventh IEEE Int'l Conf. Mobile Data Management, 2006.

[6]. 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

[7]. X. Gao and A. Hurson, "Location Dependent Query Proxy," Proc. ACM Int'l Symp. Applied Computing, pp. 1120-1124, 2005. 1120-1124, 2005.

[8]. K.C. Lee, J. Schiffman, B. Zheng, and W.-C. Lee, "Valid Scope Computation for Location-Dependent Spatial Query in Mobile Broadcast Environments," Proc. 17th ACM Conf. Information and Knowledge Management, pp. 1231-1240, 2008

[9]. K.C.K. Lee, W.-C. Lee, H.V. Leong, B. Unger, and B. Zheng, "Efficient Valid Scope for Location-Dependent Spatial Queries in Mobile Environments," J. Software, vol. 5, no. 2, pp. 133-145, Feb. 2010.

[10]. S. Prabhakar, Y. Xia, D.V. Kalashnikov, W.G. Aref, and S.E. Hambrusch, "Query Indexing and Velocity Constrained Indexing: Scalable Techniques for Continuous Queries on Moving Objects," IEEE Trans. Computers, vol. 51, no. 10, pp. 1124-1140, Oct. 2002.

[11]. Y. Cai, K.A. Hua, and G. Cao, "Processing Range-Monitoring Queries on Heterogeneous Mobile Objects," Proc. Fifth IEEE Int'l Conf. Mobile Data Management, pp. 27-38, 2004

[12]. H. Hu, J. Xu, and D.L. Lee, "A Generic Framework for Monitoring Continuous Spatial Queries over Moving Objects," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 479-490, 2005.

[13]. X. Xiong, M.F. Mokbel, and W.G. Aref, "Sea-Cnn: Scalable Processing of Continuous k-Nearest Neighbor Queries in SpatioTemporal Databases," Proc. IEEE Int'l Conf. Data Eng., pp. 643-654, 2005.

[14]. X. Yu, K.Q. Pu, and N. Koudas, "Monitoring k-Nearest Neighbor Queries over Moving Objects," Proc. 21st Int'l Conf. Data Eng., pp. 631-642, 2005.

[15]. K. Mouratidis, D. Papadias, S. Bakiras, and Y. Tao, "A Threshold-Based Algorithm for Continuous Monitoring of k Nearest Neighbors," IEEE Trans. Knowledge Data Eng., vol. 17, no. 10, pp. 1451-1464, Nov. 2005.

[16]. M.A. Cheema, Y. Yuan, and X. Lin, "Circulartrip: An Effective Algorithm for Continuous Knn Queries," Proc. 12th Int'l Conf. Database Systems for Advanced Applications, pp. 863-869, 2007.

[17]. N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R_-Tree: An Efficient and Robust Access Method for Points and Rectangles," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 322-331, 1990.

[18]. F. Aurenhammer, "Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure," ACM Computing Surveys, vol. 23, no. 3, pp. 345-405, Sept. 1991.

[19]. B. Zheng, J. Xu, W.-C. Lee, and D.L. Lee, "Grid-Partition Index: A Hybrid Method for

Nearest-Neighbor Queries in Wireless Location-Based Services," The VLDB J., vol. 15, no. 1, pp. 21-39, Jan. 2006.

[20]. J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D.L. Lee, "Location-Based Spatial Queries," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 443-454, 2003

