I.J. Information Technology and Computer Science, **2010, 2, 1-8**
Published Online December 2010 in MECS (http://www.mecs-press.org/)

A Proxy based Framework for Efficient Range Query Processing in a Cellular Network

Muhamed Ilyas
Research Scholar, School of Computer Science
Mahatma Gandhi University
Kottayam, Kerala, India
muhamed.ilyas@gmail.com

Vijayakumar
Professor, School of Computer Science
Mahatma Gandhi University
Kottayam, Kerala, India
vijayakumar@mgu.ac.in

Abstract—**Within the last decade, the continuous development of mobile technology has motivated an intense research in mobile data services. Among these services, location-based services and location-dependant queries have attracted a lot of attention from the software industry as well as from researchers. Although a number of locations based service methods have been proposed, the performance is not satisfactory in the case of dynamic data management problems that introduce high overhead in update and query operations. In this paper, we introduce a proxy based distributed dynamic data management architecture for location based services that reduces network cost with required quality of service (QoS). The system proposes a distributed hierarchical database structure with collocated proxy for efficient service delivery to location-dependant queries especially Range Queries (RQ). The system that we propose have the following advantages. i) It performs efficient processing of location dependant queries, especially Range Queries spanning over multiple network cells ii) It reduces wireless communication between mobile devices and servers. iii) The system is scalable where location data is distributed and arranged hierarchically. We use proxy objects to carry out processing task for each mobile user. Thus proxies play an important role and in charge of routing the queries to different (distributed) databases and present the results to queries efficiently. Simulation study shows the effectiveness of our proposed architecture.**

Index Terms—**Mobile computing; location-based services; Dynamic Data Management; Range Queries; Proxy Mobile Services.**

I. Introduction

With the current advances of mobile computing technology, we are witnessing an explosion in the development of applications that provide mobile users with a wide range of services. Among these applications, location based services (LBS) and Location Dependant Information Services (LDIS) have been identified as one of the most promising area of research and development. Monitoring and evaluating location queries over static and moving objects is a fundamental functionality for many mobile location-based applications. They range from location based emergency response, fleet management, cargo tracking, child care, to location-based advertisement and location-based entertainment. A key characteristic of the location based services is that the same service request may need to be answered with completely different results as the user changes his/her location or the target move [15]. To satisfy the service request effective dynamic data management strategies are required. In this paper we are mainly concerned with the dynamic data management of LBS for servicing Range Queries. Various data management schemes for LBS have already been developed for access efficiency [3]. Access efficiency refers to how fast a request is satisfied when it accesses the data of interest. However in most of these schemes the data are served from the central server and it increases the network load and overhead in central server.

A. Distributed Query Processing

While the inherent distribution of the objects themselves suggests a distributed approach for the query processing, many existing works relies on centralized server for query processing.[5], [7]. It would be better to have a distributed data management method to keep the locations of all moving and static objects (there could be millions of objects distributed over a large geographical area) for system scalability, performance and access efficiency. While some other works such as MobiEyes [6] advocates distributed query processing, they suffer several disadvantages. Most of these methods require cooperation from the moving objects to perform efficient query processing. Moving objects must update their location to the server according to a certain policy. Therefore these approaches cannot be used when it is not possible to set an update policy on the moving objects or when the objects do not explicitly communicate with the server [9]. Also their approach may overload the user devices and increase the wireless communication overheads.

In our pervious paper [14] we presented a distributed architecture of hierarchical databases with collocated service proxies. It consists of collection of database servers, base stations, application servers and a large number of static and mobile objects. Each base station has its own coverage area called cells. It considers a distributed environment where the information about the static and mobile objects inside the cell is maintained by the database server of each cell. The database server manages the location data of mobile users and static objects, such as gas stations, restaurants, etc. Location queries are registered and evaluated at the database server. We would like to stress that we are not concerned about the problem of how a database server is aware of the location of static or mobile objects; for example, it could receive location update from GPS-equipped moving objects or it could explicitly locate the moving objects when needed.

This architecture uses dynamic data management strategies to address different queries especially, range queries, where the range overlaps with the neighboring cells. When a user issues a query to the database server, it creates a service proxy object for this user, if there is no proxy already exists for this user. It assumes that the proxy is location-aware with GPS enabled devices. The service proxy performs tasks such as accepting data request to access mobile services on behalf of the user, convert query into standard SQL statements, calculate the spatial intersection of the query scope with the neighboring cells, search the local database for the cached result that matches the request, forwarding the query to the higher level if the query scope (range) overlaps with the scope of the neighboring cells., formatting the data request to various application formats, and forwarding the result to the mobile user. The personal proxy, that is, the proxy created on a per-user basis, is a “mobile proxy” which moves with the user during location handoffs to minimize the network signaling cost while maintaining the required quality of service. The leaf nodes of the tree structure are the local database of each cell, and these databases contain the information about the data objects of that cell.

The rest of the paper is organized as follows. Section 2 provides a survey is related issues and research work. Section 3 presents classification of query. Section 4 explains the proposed architecture, proxy behavior and hierarchical database model. We also model the data dissemination method. Section 5 presents the simulation results. Section 6 concludes the paper.

II. RELATED WORKS

LBS are services that answer the mobile user queries based on the location where the queries are issued. Effective dynamic data management strategies are required for servicing such queries with reasonable accuracy and response time. With respect to the previous works of Dynamic data management and proxy-based location services, Joshi and Brewer et al. [13] discussed server-side proxies concerned with content transformation and adaptation, each acting like a gateway to interact with a number of clients. Baoshan Gu and Ing-Ray Chen [3] investigated the concept of location-aware mobile service management scheme based on personal proxies. But this scheme does not address the problem of range queries where the range of the query overlaps with neighboring cells, or partially out of the cell, especially when the mobile is at the edge of the outer cell. Also the data are served from the central server and so, the overhead in the central server is quite high [9]. To reduce the overhead in central server, several caching schemes have been proposed [5, 11]. The issue of dynamic data management in Mobile environment has also been discussed in the context of replicated database [16]. Some works considered distributed environment for dynamic data management. Jayaputra and Taniar [11, 12] focused on Range queries on static objects regarding the location of the mobile user. They considered a distributed environment where the information about static objects is served by the base station. The limitation of the above approach is that, the system has to fetch data from each neighboring cell, if the Range query spans the spatial boundaries of several cells (local servers), which may lead to high overhead in communication. Shiow-yang Wu and Kun-Ta Wu [15] suggested a set of dynamic data management strategies that employs judicious caching, proactive server pushing and neighborhood replication to reduce service cost and improve response time under changing user mobility and access patterns. In this, the caching technique and service management is suggested on a per cell basis, which requires more resources if the number of range queries which overlaps neighboring cells are high. Bamba and Liu [2] proposed a grid based approach, but it mainly focused on privacy rather than data management. Sergio Ilarri et. al [8] proposed distributed processing of location dependent queries using mobile agents. This model was very general, and different types of queries are processed in a very similar way. Therefore system can miss the opportunity to optimize different aspects that are specific to certain queries and situations.

Our work considers distributed dynamic data management using hierarchical database collocated with per user proxy. Our objective is to improve the access efficiency and reduce the network cost. We propose a hierarchical database organization and data dissemination method to reduce the cost for location based services.

III. QUERY CLASSIFICATION

Based on the analysis of the information service applications and their typical usage patterns, the data and service request (queries) can be classified into different categories [15]. The nature of the queries and results can be location independent such as Stock Pro or location dependent such as the nearby restaurant within a given range. Similarly, some data are locally available where as some are available from central or remote source. To make our design general and flexible, we assume an object based service environment. The service request (query) for information can be categorized as follows [15].

- Object Queries, that targets specified objects
- Range Queries, that targets objects located within a range restriction
- Map Queries, for area map information

A. Range Queries

In this paper, we consider only range queries. Range queries are location dependent queries to locate desired type(s) of object(s) within a specified distance around the client. Range Query is formally defined as

$$\mathbf{RQ\ (p) = \{O|O.G \cap P.G \neq \ \}}$$

where P is the query polygon, O is the reference object or the mobile user who issues the query, O.G is the geometry of object O, P is the target object and P.G is the geometry of target object P. The query returns all target objects that falls within the range of the query.

Range Queries can be classified into three based on the motion properties of O (Querying object also called reference object) and P (target object). The first class of queries consists of moving (reference object) queries over static (target) objects. A query such as “Provide locations of all gas stations within 10 miles, selling gas at less than three dollars per gallon, over the next hour” belongs to this particular class of queries. The focal object of the query is the car moving on the highway which poses this query. On the other hand a query such as “Provide the locations for all cabs within five miles of my current location” is a static query over moving objects. A query such as “Locate all customers who are looking for cabs and are within five miles of my position over the next 20 minutes” is an example of a moving query over moving objects. The cab which registers this query is the focal object of the query in this scenario.

Range Queries may be presented using SQL like syntax as below

```
SELECT projections
FROM set-of-objects
WHERE Boolean-conditions
```

where *projections* is the list of attributes that we want to retrieve from the selected objects, *set-of-objects* is a list of object classes that identify the objects intersecting to the query, and *Boolean conditions* is a Boolean expression that select objects from those included in *set-of-objects* by restricting their attribute values or demanding certain location dependent constraints. For example, ‘Provide locations of all gas stations within 10 miles, selling gas at less than three dollars per gallon’ may be presented as

```
SELECT gas-station-name, street-name
FROM gas-stations
WHERE inside (10 miles, O1, gas-stations)
        and (Rate < 3);
```

the *inside (10 miles, O1, gas-stations) and (Rate < 3)* represents the location dependent-constraint defined in our system. This may also be defined by a function *Distance (ref-object, target-object)* which obtains the distance between reference object (Querying Object) and target object, depending upon the underlying Database Management System used.

Range queries are serviced by the local database servers depending on the range of the query. If the query scope (range) falls within the current cell, then such a query can directly be processed by the local database server. If the range overlaps with the neighboring cells, supporting service requests requires neighboring base stations (local servers) to exchange information on target object. Different methods were proposed, like neighborhood replication, to address such service requests [15]. Our system uses a hierarchical tree like database structure, built with local servers to address such range queries.

IV. SYSTEM ARCHITECTURE AND RANGE QUERY EVALUATION

For simplicity, the mobile communication system is modeled as homogeneous rectangular cells [11, 12] as shown in Figure 1. Each cell is assumed to have at least one Base station (BS). Several cells are connected together to form a location area with a Base Supporting Station (BSS). The mobile client within a cell communicates with the BS of the cell through a wireless channel.

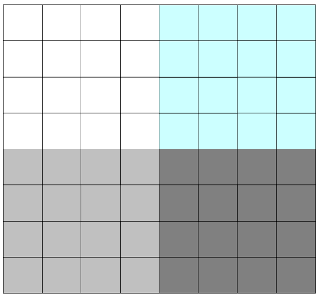


Figure 1. Regions representing location areas (LA) and individual cells (in this figure there are 4 LA, each consisting of 16 cells)

A. Database Server Hierarchy

In our previous paper [14], we devised a general service request model for location based services. Under this scheme, the local server is the data manager and wireless information server for a single cell. Each cell is assumed to have a unique local server which provides services to all clients. It maintains a local information database and a per user proxy. It is responsible for managing local information as well as the key player to provide location based services. Figure 2, shows the hierarchical structure of the cells and the databases. Four cells are grouped under two Location Areas (The number of cells is reduced for simplicity). Cell 1 and Cell 2 grouped under Location area 1 and Cell 3 and Cell 4 grouped under Location Area 2. At each server a data Management System module is in charge of storing and providing location data about static and mobile objects within the cell. It is assumed that each internal node (database) contains the cumulative information of its sub-nodes. Proper synchronization methods have to be devised for maintaining up-to-date information in all databases

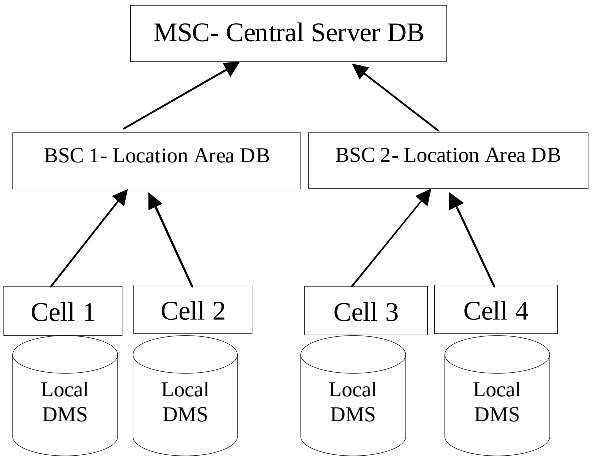


Figure 2. Hierarchical Database structure

B. *Proxy Behavior*

The proxy object plays an important role in our dynamic data management system. When a user wants to submits a query to the DMS of the local server, server creates a proxy object for that user. Then, the proxy will perform tasks such as accepting service request (query) from the client, analyzing the scope of the query, routing the request to the local or parent servers according to the scope of the query and returning the results to the client. The proxy cooperates with the underlying location management system, so that it is location-aware, and knows the location of the mobile user when the client submits a query. It is also assumed that the proxy is aware of the scope of the current cell. All service requests from the client are routed through the proxy to the database.

Proxy has the following components

1. Request Handler
2. Query Analyzer
3. Query Generator
4. Filter Engine

The main steps involved in each component is explained in the following section

C. *Query processing and Data Dissemination Model*

In this section, we explain the main steps of our query processing approach**.**

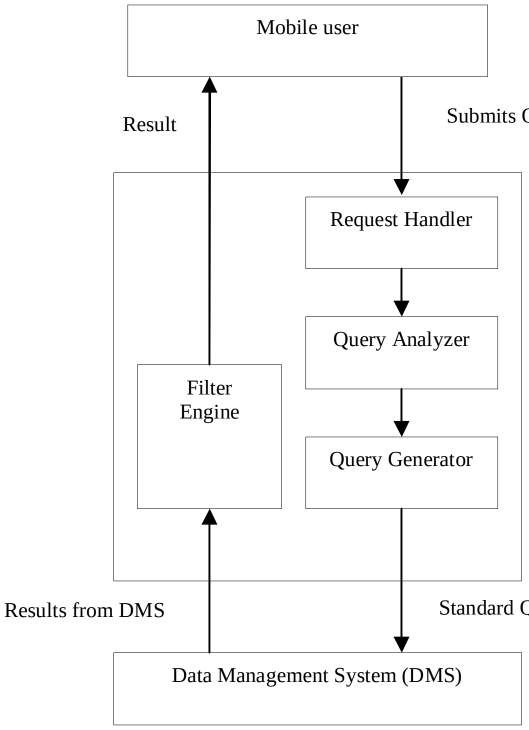


Figure 3. Main steps involved in Proxy

The user device contains an application that is used to issue queries as well as keeps the retrieved data up to date. When the user issues a query, it creates a user proxy agent at the local server collocated with local database and the query is processed according to the following steps as shown in Fig 3.

Request Handler receives the query and transforms the query to intermediate specifications in the following way i) It obtains the location of the querying object (reference object ii) It obtains the target class objects from the query iii) It also obtains the time and maximum speed of the reference object. This is useful for calculating the buffer area that can be included in the result set based on a predictive calculation. However, we are not considering these parameters in this paper.

Query Analyzer handles two things. First it identifies the spatial area of the query from the location dependent constraints of the query. During this step it checks whether the query boundary overlaps with the neighboring cells. Each local server and DMS is aware of its location area and location area of its neighboring cells. If the query boundary is completely within the cell, it is executed by the local server. If the boundary intersects with neighboring cells having the same location area, then it is routed to the next higher level. If the cells are having different location area then it is passed to the topmost level as shown in Fig 2**.** The information about the boundary intersection and query specification is forwarded to the query generator.

The query generator submits the query to the appropriate DMS based on the information obtained from Query analyzer along with the query specifications. It generates appropriate type of standard queries suitable for the DMS. (Spatial query for spatial database or standard SQL queries for relational databases). It also selects the DMS, to submit the query as we described before. If the query boundary falls within the cell, then the query is submitted directly to local DMS, otherwise it is routed to the higher levels with the information given by Query Analyzer.

The result that contains results of objects from the buffer area, based on the predictive calculation. However in this paper, we are not using the predictive calculation. Filter engine filters out the result from the buffer area and only relevant result is presented to the user.

D. Algorithm

To illustrate the service request handling procedure, we can refer to the example hierarchical database shown in fig. 2 and Fig. 4. Cell 1 and Cell 2 having the same location area and cell 3 and cell 4 is in another location area. The scope (range) of the query Q1 falls within cell 1. So this query is processed by the DMS within that cell. The scope of the query Q2 spans within Cell 3 and Cell 4. Since both these cells having the same location area, the query is routed to the upper level and processed by BSC1- location area DMS. The range of the query Q3 spans within Cell 2 and Cell 4. Since these cells are under different location area, the query is routed to MSC-the central Server DMS.

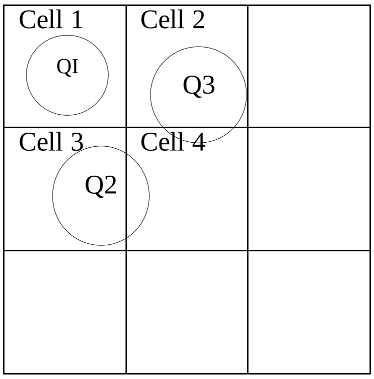


Figure 4. Range Query behavior

The following algorithm shows the proxy and database behavior and service request. Please refer Fig 5. It is supposed that the mobile client A in cell 1 issues a range query Q1(s, d, x, y, t) where s is the query string, r is the radius of the query, x and y are the geometrical coordinates available from GPS receiver attached with the mobile client and t is the time of request. The request is accepted by the proxy in the local server of cell A, then the Proxy analyze the following things

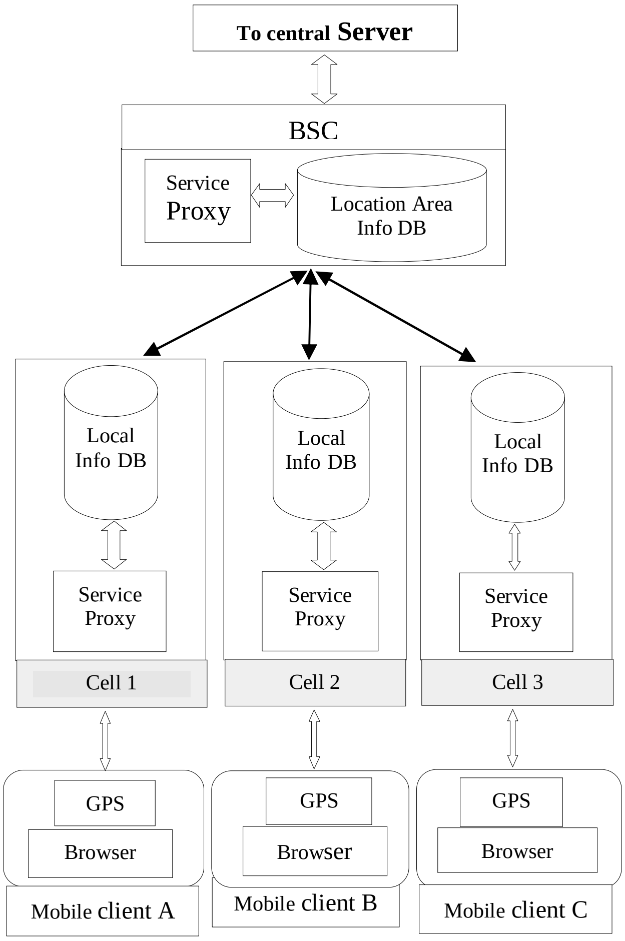


Figure 5. System Architecture for Location Based Service

- The scope (range) of the query RQ
- The category of the target data objects O

As we discussed earlier, the range of the query Q may or may not overlap the neighboring cells, according to the size of the query range (scope) and the location of the user where the query is issued. Proxy analyzes the spatial intersection of the query with the neighboring cells and works as follows

- User submits a Range Query RQ then; proxy checks the range of the query. If the range falls within the boundary of the current cell (like Q2), request is forwarded to the local server of the current cell.
- If RQ and the range overlaps with the neighboring cells (like Q1 overlaps Cell A and B), the request is forwarded to the BSC server. The BSC server maintains the copy of all data objects that are kept in local databases under its hierarchy.
- If the range overlaps with the cells under different BSC, then the request is forwarded to the higher level server, which maintains all data objects of BSC Servers under its hierarchy.

In our Example, we have shown only three level tree structure. But this can be extended to any level, based on the data management strategy.

The following algorithm shows the proxy and database behavior for service request.

Algorithm 1: Overview of proxy and database behavior

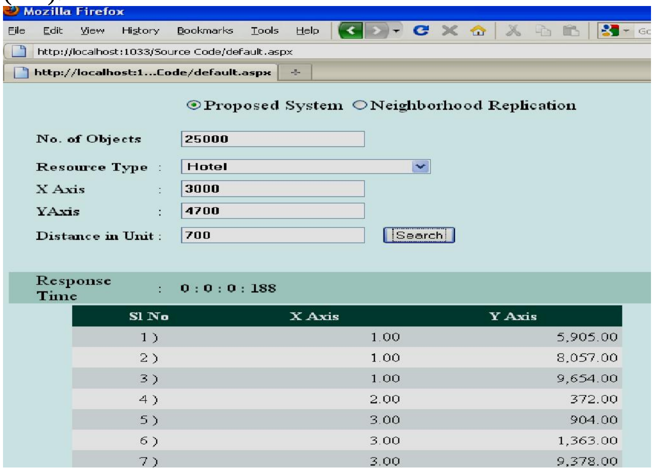
RQ	: Range Query
OC	: Object Category
d	: Distance
x	: X Coordinate of the query location
y	: Y Coordinate of the query location

```
Procedure:
While receiving a RQ do
    $BS_{scope}$ = Current Base Station Scope
    BS[8] = Neighboring Base Stations
    $Query_{scope}$ = Calculate the scope of the query from the
              values of x, x and d
if Intersection ($Query_{Scope}$, $BS_{Scope}$) is true then
    Forward the request to the parent node server
    Process the query at the parent node server
    Return the result to the client through proxy
else if query range overlaps the neighboring cell then
    Forward the request to the collocated database
    Process the query at the collocated database
    Return the result to the client through proxy
    Repeat step 11 through 15
End While.
Stop.
```

V. SIMULATION

We set the simulation with variable size of data set and partitioned the location domains using rectangular grids with varying resolution. The simulation was developed with C# and it was conducted on Intel machines with Windows 2003 as server and XP as clients.

The simulation database contains records of the random number of x, y. The number of objects for each cell is varied from 5,000 to 25,000. The experiment was conducted for Normal and Gaussian distribution. The query intersection was tested for two, three and four neighboring cells having same and different location area (LA).


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:1033/Source Code/default.aspx
http://localhost:1...Code/default.aspx
Search

● Proposed System ○ Neighborhood Replication

No. of Objects 25000

Resource Type : Hotel

X Axis : 3000

YAxis : 4700

Distance in Unit : 700

Response Time : 0 : 0 : 0 : 188

Sl No	X Axis	Y Axis
1)	1.00	5,905.00
2)	1.00	8,057.00
3)	1.00	9,654.00
4)	2.00	372.00
5)	3.00	904.00
6)	3.00	1,363.00
7)	3.00	9,378.00

Figure 6. The simulation interface

We evaluated the performance difference of our system with neighborhood replication under different objects distribution, varying number of cells and varying number of objects. Fig. 6 shows the screen of the simulation window.

Fig. 7 shows the response time under Normal distribution of objects in each cell and the query range intersection with two neighboring cells under the same location area (LA). Fig. 8 shows the response time under Normal distribution of objects in each cell and the query range intersection with four neighboring cells under the same location area (LA)

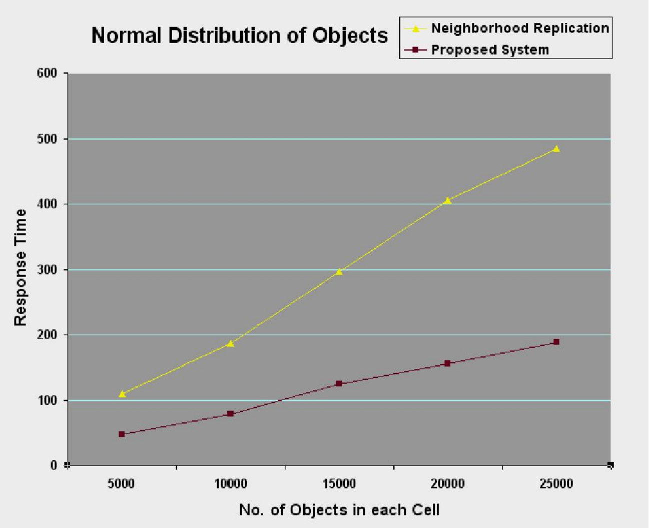


Figure 7. Response time under Normal distribution of objects and query intersection with two neighboring cells

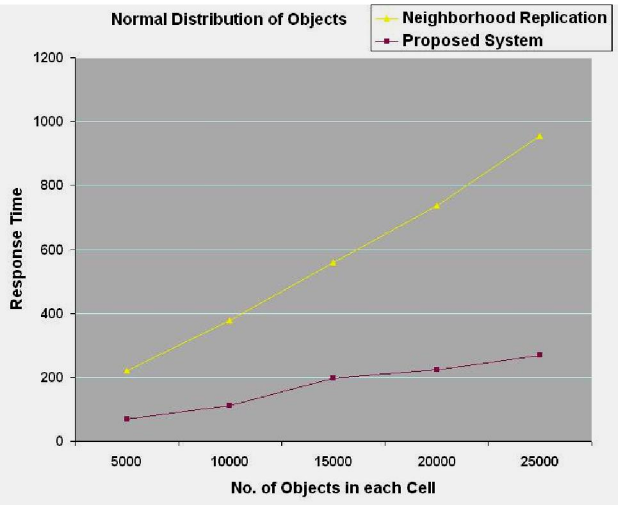


Figure 8. Response time under Normal distribution of objects and query intersection with four neighboring cells

Fig. 9 shows the response time under Gaussian distribution of objects in each cell and the query range intersection with two neighboring cells under the same location area (LA). Fig. 10 show the response time under Gaussian distribution of objects in each cell and the query range intersection with four neighboring cells under the same location area (LA)

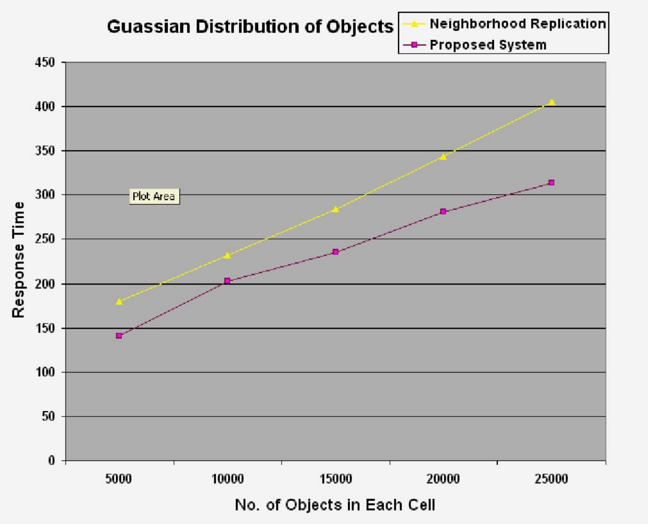


Figure 9. Response time under Gaussian distribution of objects and query intersection with two neighboring cells

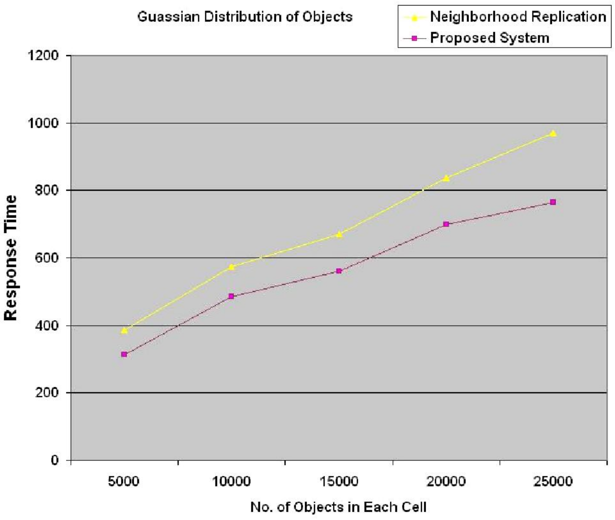


Figure 10. Response time under Gaussian distribution of objects and query intersection with two neighboring cells

The X Values are the number of objects in a cell and Y values are the response time. Under all conditions our system shows improved response time.

VI. CONCLUSION

In this paper, we proposed a hierarchical distributed database structure with collocated proxy, to facilitate location based service delivery for Range Queries. As per the architecture, each cell contains a local database, which keeps the information regarding the static objects within the cell. The Cumulative data of all cells in location area (LA) is kept in a database under BSC and the hierarchy continues upward.

A per user proxy is attached with the local database of each cell. When the user communicates with the proxy, it calculates the query scope and routes the query to the appropriate level of the hierarchical database structure, based on the spatial intersection of the query scope with the boundaries of the cell. The result is routed back to the user through the proxy. Simulation shows the effectiveness of our architecture. We are planning to extend our approach to deal with moving objects where target objects move within multiple base stations as our future work.

REFERENCES

[1] Baihua Zheng, Jianliang Xu, Wang-Chien Lee and Dik Lun Lee, "Grid-partition Index: a hybrid method for nearest-neighbor queries in wireless location-based services," The VLDB Journal Vol. 15(1), Jan 2006, pp. 21-39, doi:10.1007/s00778-004-0146-0.

[2] Bamba, B., Liu, L., Iyengar, A., and Yu, P., "Distributed Processing of Spatial Alarms: A Safe Region-based Approach," in IEEE ICDCS, pp. 207–214, 2009

[3] Baoshan Gu and Ing-Ray Chen, "Performance Analysis of Location-Aware Mobile Service Proxies for Reducing Network Cost in Personal Communication System," Mobile Networks and Applications, Vol. 10(4), Aug 2005, pp. 453-463.

[4] B.Y. Chan, A. Si and H.V. Leong, "Cache Management for Mobile Databases: Design and Evaluation," Proc.of 14th ICDE, Feb 1998, pp. 54-63.

[5] Cai. Y., Hua, K.A., Cao, G, and Xu. T, "Real-time processing of Range Monitoring Queries in heterogeneous mobile databases", IEEE Trans. Mob. Comput. 5,7 (July 2006), 931-942.

[6] Gedik, B. and Liu, L, "A distributed location monitoring service using moving location queries", IEEE Trans. Mob. Compt, 5,10 (Oct 2006), 1384-1402

[7] Hu et al. "A generic framework for monitoring continuous spatial queries over moving objects", SIGMOD Conference, 2005, pp. 479-490.

[8] Y. Huang, A.P. Sistla and O. Wolfson, "Data Replication for Mobile Computers," SIGMOD Conference, May 1994, pp. 13-24, doi:10.1145/191839.191845

[9] Ilarri, S., Mena, E., Illarraamendi, A. "Location-Dependant Query Processing: Where We Are and Where We Are Heading," ACM Comput. Surv. vol. 42.3, Article 12, Mar. 2010, pp. 2127-2130, doi:10.1145/1670679.1670682

[10] R. Jain, Y.B. Lin, C.Lo and S. Mohan, "A Caching strategy to reduce network impacts of PCS", IEEE Journal on Selected Areas in Communications , Vol. 12(8), 1994, pp. 1434-1444

[11] J. Jayaputera, D. Taniar, "Defining Scope of Query for Location-Dependant Information Services" LNCS, Springer-Verlag, Vol. 3207, 2004, pp. 23-43, doi: 10.1007/978-3-540-30121-9_35.

[12] J. Jayaputera, D. Taniar, "Data retrieval for location-dependent queries in a multi-cell wireless environment," IOS Press, Journal of Mobile Information Systems, Vol. 1, pp. 91-108.

[13] A. Joshi, "On proxy agents, mobility, and web access," ACM Journal on Mobile Networks and Application, Vol. 5(4), Dec 2000, pp. 233-241, doi: 10.1023/A:1019120915034.

[14] Muhamed Ilyas, R. Vijayakumar, "A Proxy Based Dynamic Data Management using Hierarchical Database for Location Based Services," Proc. International Conference & Workshop on Emerging Trends in

[15] Technology 2010 (ICWET 2010), Feb. 2010, pp. 75-80, ACM ISBN: 978-1-60558-812-4.

[16] Shiow-Yang Wu and Kun-Ta Wu, “Effective Location Based Services with Dynamic Data Management in Mobile Environments,” Wireless Networks Vol. 12, May 2006, pp. 369-381, doi:10.1007/s11276-005-5280-0.

[17] O. Wolfson, S. Jajodia and Y. Huang, “An adaptive data replication algorithm,” ACM Trans on Database Systems Vol. 22(2), Jun 1997, pp. 255-314, doi:10.1145/249978.249982.

Author Biographies

Muhamed Ilyas received his MCA degree from University of Kerala, Trivandrum, India. He is currently pursuing his PhD degree in the School of Computer Science, Mahatma Gandhi University, Kottayam, Kerala, India, under the guidance of Dr. R. Vijayakumar. His research interests include next-generation wireless system architectures, design and evaluation of location and service management schemes in mobile computing environments, and mobile database systems. He is a member of the ACM

Dr. R. Vijayakumar, working as Professor in the School of Computer Science, Mahatma Gandhi University, Kottayam, Kerala, India had completed 25 years of teaching career. His graduation is in B.Sc (Electrical Engineering) from College of Engineering, Trivandrum in 1984; and his M.Tech Degree in Computer Science & Engineering is from IIT Bombay in 1992. He was awarded the first Ph. D in Computer Science and Engineering in the faculty of Engineering from University of Kerala in 2000. Published 47 papers in National levels, International levels and journals in various fields of Parallel Distributed Processing and applications He is a consultant of Co-operative bank/Govt. institutions computerization and banking software/Hardware. He was member of Board of studies for Under Graduate and Post graduate studies in Kannur University and Mahatma Gandhi University, Kerala. Being the Senate Member of Calicut University, Kerala he is Chairman of Board of examiners in Computer Science and related subjects in Universities in South India. Completed sponsored projects of Govt. of Kerala and institutions in Kerala in the fields of Computer Network and applications. Apart from these he is research guide in Mahatma Gandhi University, Kerala and Anna University, Coimbatore, India.