International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume-2, Special Issue 2 (July-Aug 2014), PP. 11-14

A CACHING SCHEME FOR NEAREST-NEIGHBOUR QUERIES IN MOBILE DATABASE

Sandi Winn Aye, Zar Zar Wint
Department of Information and Communication Technology
University of Technology (Yatanarpon Cyber City)
Mandalay, Myanmar

***Abstract*—The advances in wireless and mobile computing allow a mobile user to perform a wide range of application although there are some limitations in non-mobile hard wired computing environments. As the geographical position of a mobile user is becoming more traceable, users need to pull data which are related to their location, perhaps seeking information about unfamiliar places or local lifestyle data. Respect to the limitations in mobile environments, it is strongly recommended to minimize number of connections and volume of data transmission from the servers. Caching data that are frequently accessed by mobile user in the client side is an appropriate technique for reducing data transmissions in mobile environment. Most of the existing location services providers give more than one nearest-neighbour queries results. In this paper, a new caching scheme which is based on hybrid and semantic caching is proposed to get the only one nearest neighbour query result (e.g., restaurants and gas stations) in the client side. In server side, the existing kd-tree index is used to answer nearest-neighbor queries directly.**

***Index Terms*—Hybrid caching, KD-tree, Location-dependent query, Mobile database system, Nearest-neighbor (NN) search, Semantic caching.**

I. INTRODUCTION

Wireless technology is growing rapidly and its beneficial applications which cause use of PDAs, laptops, cell phones and etc. to access data anywhere and anytime, are very common nowadays [1]. Mobile database is such a technology which confronts with some new problems, limitations and challenges (e.g., bandwidth limitations, missing connectivity, unreliable and asymmetric links) [2]. Moreover, in a mobile environment, upstream queries (i.e., from client to server) are more resource-consuming than the downstream queries (i.e., from server to client). Therefore, there is a need to reduce the number of trips made to the server [2]. Caching is becoming to be suitable and profitable in mobile environment [1]. There are several types of queries [3]. In this paper, the efficient processing of location-dependent queries and, in particular, a sub-class of queries called mobile nearest-neighbour (NN) search are focused on.

A mobile NN search is issued by a mobile client to retrieve stationary service objects nearest to its user [4]. It is an important function for location-based services (LBSs), but the implementation is difficult since the clients are mobile and queries must be answered according to the client's current locations. For example, "find the nearest restaurant" would return totally different answers to the same user when the query is issued at different locations. If a user keeps moving after a query is submitted, the problem becomes more complex because the user's current location is changing continuously and thus the results would change accordingly [3].

In this paper, a kd-tree index is used to support searching nearest neighbor data object. In addition, to enhance access efficiency of the system, a new caching scheme in which hybrid caching is combined with semantic caching, is proposed. This caching scheme stores hottest data object along with its hottest attributes.

Accordingly, this paper is continued as follows: Section II describes theory background. Related work is highlighted in section III. Section IV covers the overview of the proposed system. Description of the system is presented in section V. Finally, section VI gives the conclusion.

II. THEORY BACKGROUND

A. Mobile Database System

For any mobile architecture, things to be considered are: users are not attached to a fixed geographical location, mobile computing devices: low-power and portable, wireless networks and mobile computing constraints. Moreover, there are three parties in mobile database system. They are fixed host, mobile units and base stations.

Fixed host: It performs the transaction and data management functions with the help of database servers.

Mobile units: Portable computers move around a geographical region that is a collection of mobile cells. Mobile hosts retain network connection through the support of base stations. Role of mobile hosts depend on the capacity.

Base stations: It is a two-way radio, installation in a fixed location that passes communications with the mobile units to and from the fixed hosts. It is typically used by low-power devices such as mobile phones, portable phones and wireless routers.

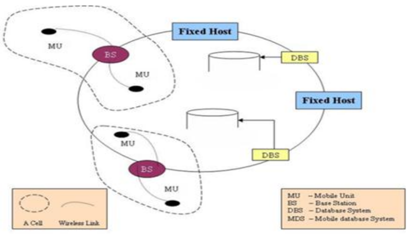


Fig. 1. Architecture of mobile database system

B. Query of Mobile Database System

In mobile environment, there are three types of entities: (i) mobile client that submits a query, (ii) mobile server that processes a query, (iii) mobile object which represents the data targeted by the query. According to these entities there are five query types in mobile environment. They are non-location related query (NLRQ), location-dependent query (LDQ), location aware query (LAQ), continuous query (CQ) and ad-hoc query.

Among these query types, in order to support location-dependent queries, some basic requirements must be met. Firstly, the user is located. And then the mobility of moving objects is maintained. Then, the future movement trends are predicted. Moreover, queries are efficiently processed and the boundaries of the precision are guaranteed.

According to the mobility of the clients and the data objects to be queried by the clients, location dependent queries can be classified into three categories: (i) mobile client querying static objects (ii) stationary clients querying moving objects (iii) mobile clients querying mobile objects.

The application area of location-dependent query can be classified into four categories: (i) location-sensitive information access (ii) object tracking (iii) location model conversation (iv) nearest-neighbor search.

In mobile computing environment, the location-dependent information for the same user at different locations is a challenging problem. Therefore, this paper focuses on location-dependent queries. Moreover, nearest neighbour search is a very important one of problems in location-dependent information services. Therefore, the performance issues of nearest neighbour search are addressed in this paper.

C. Caching

In a mobile environment, upstream queries (i.e., from client to server) are more resource-consuming than the downstream queries (i.e., from server to client). Therefore, there is a need to reduce the number of trips made to the server [2]. Caching is to be profitable in such situation. However, the caching techniques used for traditional database models cannot be applied in this area as it is [2]. Additionally, mobile clients also have limited resources like power. Therefore, any of the caching schemes have to be energy efficient and support long and frequent disconnections. Hence, efficiently caching some of the elements that are frequently required by the mobile device can save the scarce resources and give better response time for the client application.

There are four caching types: (i) Object Caching: Each mobile client tends to have its own set of hot objects that it accesses most frequently. (ii) Attribute Caching: After the server has evaluated the query submitted by client, it returns only those attributes of the qualified objects that are requested by client. Not caching all attributes saves cache space. (iii) Hybrid Caching: It is a mix of object and attributes caching. Therefore, this caching store attributes of objects that are most frequently used (iv) Semantic Caching: In this caching, the mobile client maintains the semantic descriptions and results of previous queries in its cache.

III. RELATED WORKS

Xu et al. proposed a new index structure called D-tree [8]. D-tree indexes spatial regions based on the divisions that form the boundaries of the regions. It is an object-based index and has a small index size since it only indexes the necessary information i.e., the position information of objects. However, it requires lots of backtracking in the whole search process. Lee et al. projected a method for answering location-dependent queries in a mobile computing environment [3]. This technique constructs a Voronoi Diagram (VD) on the data objects to serve as an index for data objects. Nevertheless, VD is a solution-space index and its drawback is large index size. Zheng et al. proposed a new index method to support nearest-neighbour queries [6]. Both solution-based index and object-based index have advantages and disadvantages. Therefore, advantages of both indexes are combined in this method. However, this proposed index method is planned for wireless broadcasting. Respect to limited resources in mobile environments, it needs to minimize number of connection and volumes of data transmission from the servers. Caching seems to be very profitable approach in such situation. A semantic cache is proposed to enhance the access efficiency of the service by using Voronoi Diagram [4]. Cache replacement policies for the semantic cache are examined. Several query scheduling policies are proposed to address the inter-cell roaming issues in multi-cell environments. Ali A. Safaei et al. projected a semantic cache schema for continuous k-NN queries in mobile DBSs [1]. Continuous nature of moving in addition to queries raises this fact that caching the previous k-nearest objects in client system's cache will be applicable for the forthcoming queries. The various caching mechanisms are proposed for mobile devices with the emphasis on critiquing the assumptions made in various schemes [2]. These techniques are compared based on their strengths and weakness. The wireless technology has made it possible to achieve continuous connectivity in mobile environment [5]. When the query is specified as continuous, the requesting mobile user can obtain continuously changing result. In order to provide accurate and timely outcome to requesting mobile user, the locations of moving object has to be closely monitored. The problem related to the role of personal and terminal mobility and query processing in the mobile environment are discussed.

IV. OVERVIEW OF SYSTEM

This section presents an overview of the proposed system. Taking the advantages of caching scheme, this system offers the query result to the users based on their current location quickly. Moreover, the connection and data transmission from the Database server to the client can be reduced. The system flow of the proposed system is illustrated in Fig. 2.

The proposed system consists of two sides: sever side and client side. As the pre-processing, the search space is divided by using kd-tree index in the server side. In the client side, the hybrid caching is combined with semantic caching. The previous query answers are cached in the client system's cache according to the proposed caching's policy. If there is a suitable answer corresponding to the user's query in the client's cache, this query can be answered from that cache without connecting to the server.

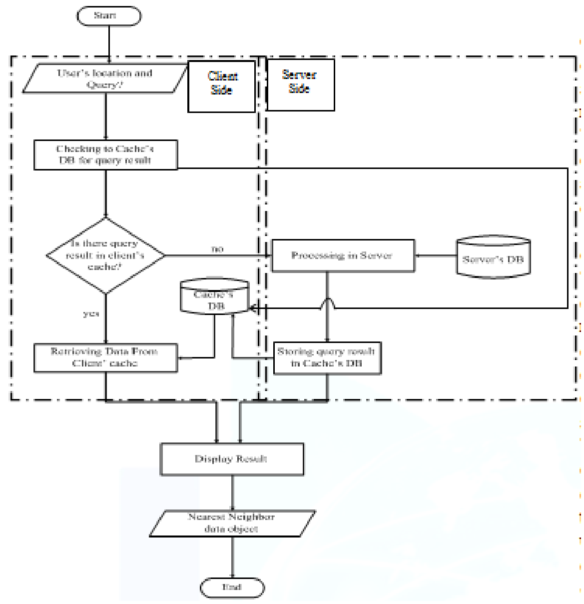


Fig. 2. Flow chart of proposed system

At first, mobile user requests query with his/her current location. And then, the client's cache is check to see whether the query answer is available. If there is a suitable cache record corresponding to the location of the user, the most appropriate answer is retrieved from the cache and this answer can be returned to the mobile user quickly. Otherwise, the current location of the user and query are transmitted to the server. The nearest-neighbour object to the user is computed in the server. Then, the result is returned to the user. After the user gets the result from the server, a new query answer is inserted in the client's cache.

V. DESCRIPTION OF SYSTEM

This section describes how to cache query result in the client side for location-dependent application according to the limitation of mobile environment and how to divide the search space and to compute nearest-neighbor service in the server side and then shows the experimental results for clarify.

A. *Client Side*

In mobile environment, the connection to the server is not always available as there are frequent disconnections. Other resources like power capabilities of mobile devices are not high. Respect to limitations in this environment, it is strongly recommended to save the trip to the server. Caching seems to be profitable in such situations. In the proposed system, a new caching scheme in which hybrid caching is combined with semantic caching for location-dependent queries (mainly for Nearest-Neighbor Queries) is proposed. Caching the previous nearest object in client system's cache will be applicable for the forthcoming queries. Nevertheless, all previous nearest objects can't be cached in client's cache because of limited size of the cache. It needs to cache some of the objects that are frequently accessed by database applications. In the proposed caching scheme, there are two types of caching: hybrid caching and semantic caching.

1) *Hybrid Caching:* Hybrid caching is a mix of object caching and attributes caching. In object caching, it stores an object that is frequently requested by the mobile user. All attributes of this object are cached in the client side. However, not all attributes of this object will be accessed in the future. This not only wastes the transmission bandwidth, but also occupies storage for caching other more frequently accessed attributes. In the attribute caching, it cache attributes of this object that is mostly accessed.

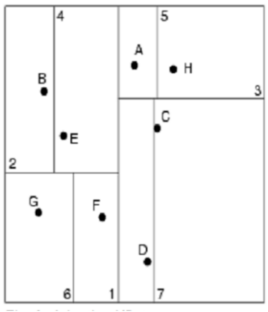
2) *Semantic Caching:* The semantic description and results of previous query results are stored in the mobile client cache. Therefore, if a new query is totally answered from the client cache, communication to the server is unnecessary. But if a new query doesn't exactly match the one from the client cache, it needs to connect with the server. By using semantic caching, a query can be partially answered from the client cache in above condition. Then the original query is trimmed and the trimmed part is sent to the server and processed there. The amount of connection and data transmission to the server can be reduced by processing queries according to this caching. Main advantage of semantic caching is that by trimming queries, previous results in the client cache can be used if the query is not an exact match. Moreover, the communication from client to server can be reduced. On disadvantages, only select and project queries can be applied. The semantic description of the proposed system contains object name, attribute set, the location of client that is described by x, y coordinate as <SO,SA,PX,PY>. This semantic description is shown in Table 1.

TABLE I. EXAMPLE OF SEMANTIC CACHE DESCRIPTION

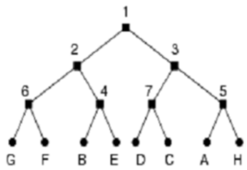
SO	SA	PX	PY
hotel	street	100	300
hotel	street,tel	200	300
restaurant	street	400	200
restaurant	cuisine	200	100

B. *Server Side*

In the server side, the search space is divided by using kd-tree index. It recursively partitions the search space into two complementary subspace such that the number of objects associated with the two subspaces is nearly the same. The space is partitioned horizontally or vertically. The point set is split alternatingly by x-coordinate and by y-coordinate. Splitting by x-coordinate: the point set is split by a vertical line that has half the points left and half right. Splitting by y-coordinate: it is split by a horizontal line that has half the points below and half above. Each internal node stores the splitting node along x (or y).



(a)



(b)

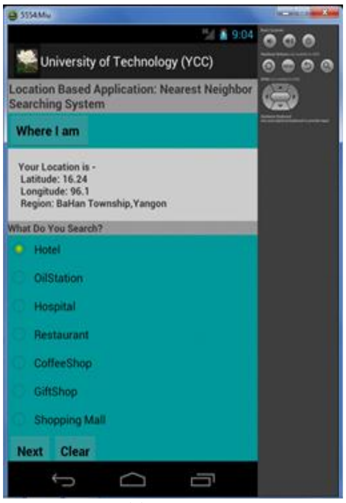
Fig. 3. KD-Tree for running example (a) Division in example (b) KD-tree index structure

In nearest-neighbor search, it starts with the root node, the algorithm moves down the tree recursively. It goes right or left depending on whether the point is greater or less than the current node in the split dimension). Once the algorithm reaches a leaf node, it saves that node point as the "current best".

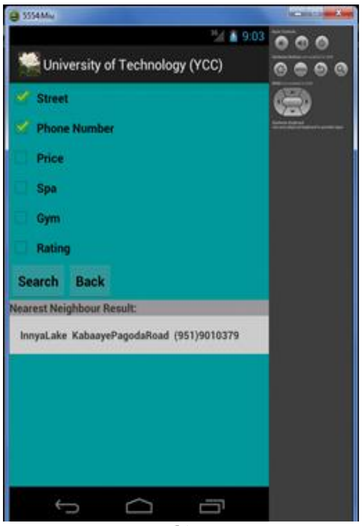
C. *Experiment*

This system considers only one nearest-neighbor result of evaluating user query based on his current location. The user looks for his current location which is shown as latitude and longitude from GPS and there are seven services which he can choose are shown in Fig. 4(a). Moreover, some attributes related to each service are shown in Fig. 4(b). Firstly, one service, which she wants to find nearest one to her current location, and it's some attributes can be chosen. After getting her location and choosing one service and its attributes, the system checks the client's cache base on her current position to see whether the query answer is available. If there is a suitable cache record corresponding to the location of the user, the most appropriate answer is retrieved from the cache and this answer can be returned to the mobile user quickly. Otherwise, the current location of the user and query are transmitted to the server. The server finds one nearest neighbor object and its attributes according to the user's current location and sends back one nearest neighbor object to the user.

As user current location shown in Fig. 4(a), his current location is in BaHan Township of Yangon Region. And then, he wants to find nearest-neighbor hotel including its street name and phone number. Therefore, he chooses hotel and its attribute (street, phone Number). Eventually, the system extracts one nearest neighbor hotel (InnyaLake Hotel) to user current location and then the only one nearest-neighbor result with related attributes is returned to the user as shown in Fig. 4(b). If the result is got from the server, this new query result is inserted in the client's cache. The experimental results are shown in Fig. 4.



(a)



(b)

Fig. 4. Client terminal display (a) user current location from GPS and services (b) attributes and one nearest-neighbor result from the server

VI. Conclusion

In this paper, the location-based application that provides only one nearest neighbour based on user current location has been discussed. To overcome limitations in the mobile environment, this paper presents that data caching is a profitable solution. A new caching scheme is proposed to find only one nearest neighbour result and its related attributes based on user current position. Hybrid caching and semantic caching are used by the proposed system for reducing wireless data transmission and cache size. Moreover, kd-tree index is used in the server side to answer queries for finding the nearest services facilities based on mobile client's locations more accurately.

Acknowledgment

I specially thank to my Supervisor Dr. Zar Zar Wint for fruitful discussion during the preparation of this paper. I would also like to thank Rector, Professors and my colleagues from University of Technology, Yatanarpon Cyber City, Myanmar.

References

[1] A.A. Safaei, M. Mallah, F. Abdi and S. Behjati, "Semantic Cache Scheme for Continuous k-NN Queries in Mobile DBS", Information Technology Research Journal, 1(1), 2011, 21-32.

[2] R. Rathore, R. Prinja, "An Overview of Mobile Database Caching".

[3] B. Zheng and D.L. Lee, "Semantic Caching in Location-Depenent Query Processing". April 3, 2001.

[4] B. Zheng, W.C. Lee, D.L. Lee, "On Semantic Caching and Query Scheduling for Mobile Nearest-Neighbour Search".

[5] S.D. Sharma, "Mobile Database System: Role of Mobility on the Query Processing", International Journal of Computer Science and Information Security, 7(3), 2010.

[6] B. Zheng, J. Xu, W.C. Lee, D.L. Lee, "Energy-Conserving Air Indexes for Nearest Neighbour Search", 2004.

[7] Q.Ren and M.H. Dunham," Semantic caching in mobile computing", 2000.

[8] J. Xu, B. Zheng, W.C. Lee and D.L. Lee, "The D-tree: An index structure for Planar Point Queries in Location-Based Wireless Services".19th IEEE Int'l Conf. on Data Engineering (ICDE '03), Bangalore, India, 2003.