Quick Retrieval of Nearest Neighbor by using Keywords

Vidya L. Tikone¹, Snehal M. Tembare², Manisha L. Narad³, Bharat V. Supekar⁴, Nilesh T. Pawar⁵
¹,²,³,⁴,⁵ U.G. Students, Department of Computer Engineering
SRES’s COE, Kopargaon, Saitribai Phule Pune University

ABSTRACT

Now days there are many modern android applications that aim to find out objects satisfying both spatial predicate and a predicate on their associated texts. In this paper, for finding nearest hotel a simple solution is introduced which relies on IR² tree [1]. IR² tree includes few deficiencies that affect its efficiency [2]. To increase the efficiency a new method called spatial inverted index [1] is introduced that extends the standard inverted index to address multidimensional information. This new SI index method comes with algorithms which will answer nearest neighbor queries with keywords in real time. As verified by experiments, this projected technique performs the IR² tree in query time reaction very significantly.

Keywords: Information Retrieval Tree, Keyword Search, Spatial Inverted Index.

I. INTRODUCTION

Current systems searches location on the basis of their geometric location from the user location. In this paper we are going to develop an application which will search the nearest location with given keywords. In this paper we considered only hotel as a location and keywords as a Hotel Menus. For example if user wants nearest hotel then he can find it with its famous dish. Means if user wants “Paneer” only then he can enter paneer as keyword then it will return nearest hotels which has paneer menu. Currently the best solution to such queries is based on the IR²-tree, which is used in this paper. This algorithm is very efficient to search location with given keywords [1]. Also there is a method called spatial inverted index that is used with multidimensional data and that comes with nearest neighbour search with given keywords.

Existing System: In the previous system, real nearest neighbor lies quite far away from the query point, while all the closer neighbors are missing at least one of the query keywords. As its fail to find the nearer location of restaurant having with all the keywords or menus available in restaurant. Existing system mainly focus on finding the nearest top neighbor where each node have to match whole query keywords. All these method are low efficient for incremental query.

Proposed System: To overcome the drawbacks of previous applications, we proposed an application for android users. In our system we are mainly dealing with searching and nearer location issues and database manage multidimensional objects which resulted in failure of previous systems [2]. To deal with spatial index as searching the entered keyword and from that find the nearest location having that keyword available and showing the location of restaurant having menus available in map. So easier to find the location of nearer restaurant in map having the available keyword.

System Design (Use Case Diagram):

Fig. 1 Use Case Diagram

Related Work: A spatial database manages multidimensional objects (such as points, rectangles, etc.), and provides fast access to those objects based on different selection criteria. The importance of spatial databases is reflected by the convenience of modeling entities of reality in a geometric manner [3]. For example, locations of restaurants, hotels, hospitals and so on are often represented as points in a map, while larger extents such as parks, lakes, and landscapes often as a combination of rectangles [4]. Many functionalities of a spatial database are useful in various ways in specific contexts [5].
II. MATERIALS AND METHODS

Module 1: Customer Registration:
In this module, the user will have to register first. Once the user does the registration then he/she can access the application. For registration user have to enter the basic information about himself. User also have to set the username and password. This all registration information is get stored into database. The IMEI number is automatically get stored into database once user do the registration.

Module 2: Customer Login:
In this module, after the registration customer can login through mentioned username and password.

Module 3: Hotel Registration:
In this module, Admin register the hotel with its famous dish. Hotel owner have to do the registration then only the hotel get search through application. Also hotel owner have to add the menu which is available in the hotel so that user can search the hotel through keyword. Only registered hotels will be displayed in the application. These hotel’s location will be seen in the map with distance. Each hotel owner will get the separate username and password for login.

Module 3.1: Hotel Login / Admin:
In this model once Hotel Owner login into application then he can insert the menu or update the menu.

Module 4: Searching Keyword:
In this module, the user will enter the keyword searching for menus available in restaurant which will nearer from its position. Whenever user will enter keyword (menu name) it will match data with the hotel database server and find the nearest restaurant with the available entered menu by customer. For nearest restaurant we are using IR2Tree & compression. The IR2-Tree is a combination of an R-Tree and signature files. In particular, each node of an IR2-Tree contains both spatial and keyword information; the former in the form of a minimum bounding area and the latter in the form of a signature. An IR2-Tree facilitates both top-k spatial queries and top-k spatial keyword queries as we explain below. More formally, an IR2-Tree R is a height-balanced tree data structure, where each leaf node has entries of the form (Obj Ptr, A, S). Obj Ptr and A are defined as in the R-Tree while S is the signature of the object referred by Obj Ptr. Anon-leaf node has entries of the form (Node Ptr, A, S). Node Ptr and A are defined as in the R-Tree while S is the signature of the node. The signature of a node is the superimposition (OR-ing) of all the signatures of its entries. Thus a signature of a node is equivalent to a signature for all the documents in its sub tree.

Algorithm:
Input:-Menu list, Input query
Algorithm IR2TreeGeneration ()
{
1. Let Hl be the list of hotels along with their latitude and longitude
   Let Hm be the menu available in respective hotel
2. Let Sm be the signature set for each menu,
   For each (menu in Hl)
   {
   //signature generation
   Hashing (menu);
   Add hash menu in Sm;
   }
3. Let Sh be the signature for hotel list
   For each (Hl in List)
   {
   //Generation of signature for each Hotel
   Hashing (menu (i) OR menu (i+1));
   Add hashing to Sh
   }
4. Declare HLc is list of hotel having all menu which satisfies query
   For each (val in Sh)
   {
   Check if (q = val)
   Add hotel in HLc
   }
5. Declare SHl sorted list of hotels from HLcS
   Declare Dmin is min distance from list of sorted hotels //List along with distances
   Hl=Sorted list (HLc);
   Dmin=min(Hl)
7. Sorting
   User can search the hotel by sorting the names of hotels.

Module 5: Map view / Searching Location:
In this module, all the names of restaurant will appear in the list which came from database and find the position in map (google_play_service_library) is required for showing position of restaurant in map and which will be more easier to customer to get the nearer restaurant from its current position.

Module 6: Distance Search:
In this module, customer can find the distance from source to destination. So that it can be easier to find the distance and reached the destination. It will give the distance of the hotel from the current location.
Logic for calculating distance between user and hotel:
\[
d\text{Lat} = \text{Math.toRadians} (\text{lat1}-\text{lat2});
\]
\[
d\text{Lng} = \text{Math.toRadians} (\text{lng1}-\text{lng2});
\]
double a = Math.sin(dLat/2) * Math.sin(dLat/2)
+ Math.cos(Math.toRadians(lat2)) *
 Math.cos(Math.toRadians(lat1)) *
 Math.sin(dLng/2) * Math.sin(dLng/2);

double c = 2 * Math.atan2(Math.sqrt(a),
 Math.sqrt(1-a));

double dist = earthRadius * c;

Solutions based on inverted indexes:

Inverted indexes (I-index) have proved to be an effective access method for keyword-based document retrieval[1]. In the spatial context, nothing prevents us from treating the text description Wp of a point p as a document, and then, building an I-index. Each word in the vocabulary has an inverted list, enumerating the ids of the points that have the word in their documents.

According to the experiments of [2], when Wq has only a single word, the performance of I-index is very bad, which is expected because everything in the inverted list of that word must be verified. Interestingly, as the size of Wq increases, the performance gap between index and IR2-tree keeps narrowing such that I-index even starts to outperform IR2-tree at |Wq| = 4. This is not as surprising as it may seem. As |Wq| grows large, not many objects need to be verified because the number of objects carrying all the query keywords drops rapidly. On the other hand, at this point an advantage of index starts to pay off. That is, scanning an inverted list is relatively cheap because it involves only sequentialI/Os1, as opposed to the random nature of accessing the nodes of an IR2-tree.

III. RESULT AND DISCUSSION

The SI-index, accompanied by the proposed query algorithms [1], has presented itself as an excellent tradeoff between space and query efficiency. Compared to IFR, it consumes significantly less space, and yet, answers queries much faster. Compared to IR2-tree, its superiority is overwhelming since its query time is typically lower by a factor of orders of magnitude.

IV. CONCLUSION AND FUTURE SCOPE

Our Project Fast Nearest Neighbor Search with Keywords is extremely effective for searching nearest restaurant from user with expected menus. It does this by IR2 tree algorithm- Compression, Merging and Distance Browsing, and GPS System. In this we can add features like After selecting Hotel it will display menu card of that Hotel Implement this application for PC’s and Desktops.

We have so many applications that can be used as search engine which is able to efficiently support novel forms of spatial queries that are integrated with keyword search. In this project we have developed an access method called the Spatial Inverted Index (SI-Index). This method is fairly space economical and it has ability to perform keyword augmented nearest neighbor search in real time. This method is based on conventional technology of Inverted Index. It is readily incorporable in a commercial search engine.

REFERENCES

1) Yufei Tao and Cheng Sheng In proc of Chinese University “Fast Nearest Neighbor Search with Keywords”


