CHAPTER-2

LITERATURE SURVEY

The idea of collaborative spatial computing has been widely used in various domains including location based social networks, geo-crowd sourcing, activity planning , group decision making and disaster rescue . One of the most important applications of collaborative spatial computing in the database field is geo-social queries, which are attracting increasing interest from both industrial and academic communities.

2.1Queries used in Geo-Social Database

2.1.1 k-Nearest Neighbor Queries

Considering that k -Nearest Neighbor (k -NN) search is one of the most popular web interfaces in accessing spatial data on the web, The problem of retrieving geospatial data from the web for a given spatial range query using only k -NN searches were studied. Besides web data integration and search applications, the solution to this more general problem is beneficial for other application domains in the areas of sensor networks, online maps and ad-hoc mobile networks where their applications require great deal of data integration for data analysis and decision support queries. For example, the McDonalds web site provides a restaurant locator service through which one can ask for the five closest restaurants from a given location. This type of web interface to search for a number of “nearest neighbors” from a given geographical point (e.g., a mailing address) or an area (e.g., a zip code) is very popular for accessing geospatial data on the web. It nicely serves the intended goal of quick and convenient dissemination of business location information to potential customers. However, if the consumer of the data is a computer program, as in the case Geo informatics of web data integration utilities (e.g., wrappers) and search programs (e.g., crawlers),such an interface may be a very inefficient way of retrieving all data in a given query region.[1]

2.1.2 Nearest Neighbor Queries

A frequently encountered type of Query in Geographic Information Systems is to find the k-Nearest Neighbor objects to given a given point in space. Processing such queries requires substantially different search algorithms than those for location or range queries.[2]

2.1.3 GSKCG Queries

Given a set of spatial query points and an underlying social network, a GSKCG query finds a minimum user group in which the members satisfy certain social relationship and their associated regions can jointly cover all the query points. For the spatial factor, instead of finding a group of users close to the query points (e.g.,Spatial task sites or a rally point), a GSKCG query finds a user group whose associated regions (e.g,Service regions or familiar regions) jointly cover a set of query points; for the social factor, we employ the more reasonable k-core notion to measure the intensity of the relationships of users in the selected group, for example, each user should be familiar with at least k other users. For this reason, the techniques developed for previous geo-social queries cannot be directly applied to our problem.[3]

2.2 Data Structure used in Spatial Databases

2.2.1 Inverted spatial Index

Spatial queries with keywords have not been extensively explored. In the past years, the community has sparked enthusiasm in studying keyword search in relational databases. It is until recently that attention was diverted to multidimensional data. The best method to date for nearest neighbor search -They nicely integrate two well-known concepts: R-tree, a popular spatial index, and signature file, an effective method for keyword-based document retrieval. By doing so they develop a structure called the IR^2-tree , which has the strengths of both R-trees and signature files.

Like R-trees, the IR^2 −tree preserves objects' spatial proximity, which is the key to solving spatial queries efficiently. On the other hand,like signature files, the IR^2-tree is able to filter a considerable portion of the objects that do not contain all the query keywords, thus significantly reducing the number of objects to be examined.

The IR^2-tree, however, also inherits a drawback of signature files: false hits. That is, a signature file, due to its conservative nature, may still direct the search to some objects, even though they do not have all the keywords.

The penalty thus caused is the need to verify an object whose satisfying a query or not cannot be resolved using only its signature, but requires loading its full text description, which is expensive due to the resulting random accesses. It is noteworthy that the false hit problem is not specific only to signature files, but also exists in other methods for approximate set membership tests with compact storage .Therefore, the problem cannot be remedied by simply replacing signature file with any of those methods.

It assumes the knowledge of R-trees and the best-first algorithm for NN search, both of which are well-known techniques in spatial databases. Signature file in general refers to a hashing-based framework, whose instantiation in is known as superimposed coding (SC), which is shown to be more effective than other instantiations .

It is designed to perform membership tests: determine whether a query word w exists in a set W of words. SC is conservative, in the sense that if it says "no", then w is definitely not in W. If, on the other hand, SC returns "yes", the true answer can be either way, in which case the whole W must be scanned to avoid a false hit.

The IR^2-tree is the first access method for answering NN queries with keywords. As with many pioneering solutions, the IR^2-tree also has a few drawbacks that affect its

efficiency.The most serious one of all is that the number of false hits can be really large when the object of the final result is faraway from the query point, or the result is simply empty. In these cases, the query algorithm would need to load the documents of many objects, incurring expensive overhead as each loading necessitates a random access.[11]

2.2.1.1 The Compression Scheme

Compression is already widely used to reduce the size of an inverted index in the conventional context where each inverted list contains only ids. In that case, an effective approach is to record the gaps between consecutive ids, as opposed to the precise ids. For example, given a set S of integers {2, 3, 6, 8}, the gap-keeping approach will store {2, 1, 3, 2} instead, where the i^{th} value (i ≥2) is the difference between the $i^{th}$and $(i-1)^{th}$ values in the original S.As the original S can be precisely reconstructed, no information is lost.

The only overhead is that decompression incurs extra computation cost, but such cost is negligible compared to the overhead of I/Os. Note that gap-keeping will be much less beneficial if the integers of S are not in a sorted order. This is because the space saving comes from the hope that gaps would be much smaller (than the original values) and hence could be represented with fewer bits.

This would not be true had S not been sorted. Compressing an SI-index is less straightforward. The difference here is that each element of a list, a.k.a. a point p, is a triplet (id_p, x_p, y_p), including both the id and coordinates of p. As gap-keeping requires a sorted order, it can be applied on only one attribute of the triplet. For example, if we decide to sort the list by ids, gap-keeping on ids may lead to good space saving, but its application on the x- and y-coordinates would not have much effect. To attack this problem, they first leave out the ids and focus on the coordinates.

Working of inverted index tree is explained with the help of graph as below,

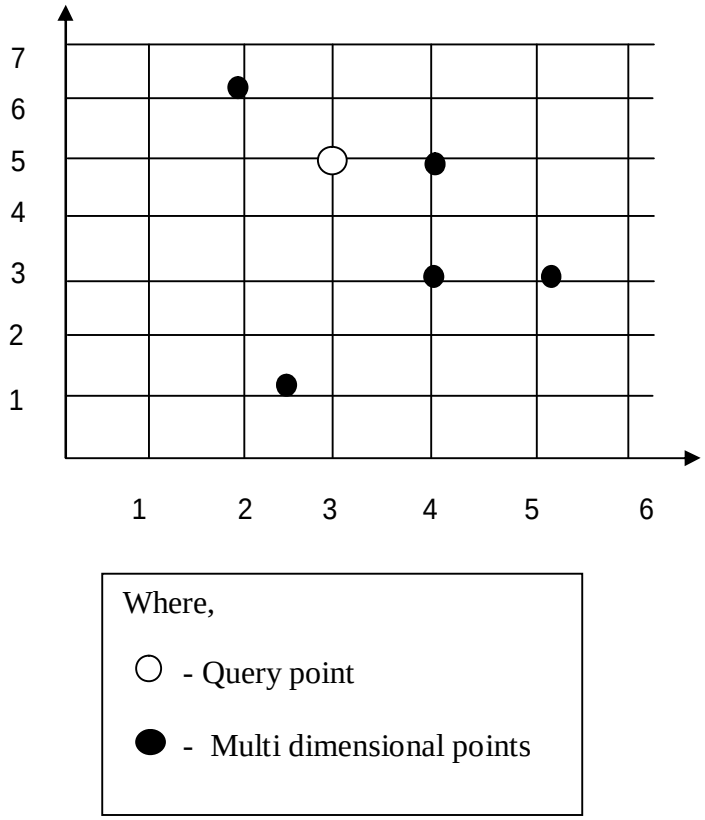

Figure.2.1 Location of Points

Let P be a set of multidimensional points. As we combine keyword search with location and textual information on facilities such as restaurants, hotels, etc. Here, we mainly focus on dimensionality 2 by considering the points in P with integer co ordinates [0, t],where t is a large integer. The valued coordinates represented in 2D is still finite and enumerable. So with proper scaling we proceed with such consideration.

Each character represented in the graph is an object in P with its textual data as a document is represented by W_p.For example, if alphabets stands for restaurant, W_p can be its menu.In addition,it may also have different useful informations.

A Nearest neighbor (NN query) gives a point q and a set W_q of keywords (W_q as the document of the query).It returns the point in P_q that is the nearest to q, where P_q is defined as,

$$\boldsymbol{P_q = \{character \epsilon P | W_q \subseteq W_p\}} \text{ ..... (1)}$$

In simple, P_q is the set of objects in P whose documents contain all the keywords in W_q. If the P_q returns is empty, the query returns nothing. This problem as an overall, can be considered as *k* nearest neighbor (*k* NN) search, which finds the *k* points, the entire P_q should be returned.

For example, assume that P consists of 6 points whose locations and a query point q are given as black dots and white dots respectively in the Figure. 2.1. Figure.2.2 describes the associated text for Characters as below,

P	$\boldsymbol{W_p}$
A	{a,b,c,d}
B	{c,g}
C	{e,g}
D	{f,g,i}
E	{a,f}

Figure.2.2 Associated text for characters

Consider the query point q at the white dot of Figure.2.2 with the set of keywords W_q={f,g}.Nearest neighbour finds D as the nearest neighbor as F misses {f} in it. If k=2,In addition,E is also returned. So the result set contains two character points namely

{D,E}.The result set remains unchanged for k=3 or higher values as they were the only two objects that have both the keywords{f,g}.

Inverted Indexes(I-index) have proved to be an effective access method for keyword-based document retrieval. Consider the Figure.2.3 below,

Word	**Inverted list**
a	A,E
b	A
c	A,B
d	A
e	C
f	D,E
g	D
i	D

Figure.2.3 Example of an Inverted Index (I-Tree)

It contains the index for the data set of Figure.2.3.Each word in the vocabulary has an inverted list by pinpointing the ids of the points that have the word in their documents. The list of each word maintains a sorted order of point ids. Thus it provides considerable convenience in query processing by allowing a merge step.

Given a nearest neighbor query q with the keyword set W_q ,the query algorithm of I-Index first retrieves the set $P_q$of the points that have all the keywords of W_q.Then it ranks

them based on the distance from the centre point. Then, our system analyses the menu items which were unique with the menu items of the resultant restaurant's list. Then it considers as how many menu items were repeated and recommends the items that are to be added for our business.

In case, if no restaurant is available with such queried menu items, then it recommends the new such menu item if we opt to start restaurant.Thus,it eliminates the time consumed for gathering information about how to develop business. Thus, the system helps the individual as well as society's development by reducing time and saving cost.

Solutions based on inverted index will not work here because, The list of each word is maintained in a sorted order of point ids, which may provide considerable convenience in query processing by allowing an efficient merge step. For example, assume that we want to find the points that have c and d. This is essentially needed to compute the intersection of 2 lists.

M-closest keywords problem is not considered here. False hits can be eliminated by the database structure thus distance is accurately calculated. A serious drawback of R-Tree approach is its space cost. Notice that a point needs to be duplicated once for every word in its textual description, resulting in very expensive space consumption for large data sets. Gap keeping may not suit for real data sets that we have in a city .They use this for spatial inverted index. But index structure increases the problem's complexity instead of reducing it.

2.2.2 IR^2 Tree

The IR^2 **Tree** is a combination of an R -Tree and signature files. In particular, each node of an IR^2 **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 IR^2 **Tree** facilitates both top -k spatial queries and top -k spatial keyword queries. More

formally, an $\boldsymbol{IR^2}$ **Tree** R is a height - balanced tree data structure, where each leaf node has entries of the form (ObjPtr, A, S). ObjPtr and A are defined as in the R - Tree while S is the signature of the object referred by ObjPtr. A non-leaf node has entries of the form (NodePtr, A, S). NodePtr 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 signature for all nodes in the sub tree.[12]