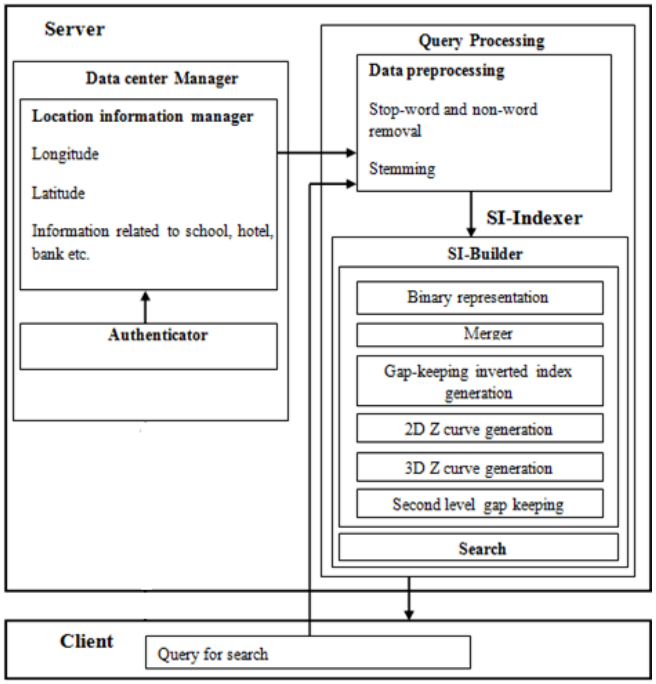

Figure 1: System architecture

A . Data center manager

In first module location information manager use "Google Map" of particular area, by selecting any location on that map it gives longitude and latitude. The retrieved longitude and latitude saved into the database.It also includes information related to school, bank, hotel, restaurant, college etc. Data preprocessing is applied on the data that is managed by information manager.

B. Query preprocessing

1. Data preprocessing

When user enter the query to search then it is passed to the data preprocessing, first preprocessing of the query is stop-words removal from that query. Stop-words are frequently occurring and unimportant words in a language that helps to construct sentences but do not represent any content of the documents. Stop-words in include: a, about, an, are, as, at, be, by, for, from, how, in, is, of, on, or, that, the, these, this, to, was, what, when, where, who, will, with. Such words should be removed before documents are indexed and stored. For example if user query is "search for nearest school" then remove "for" word from that query. Next step is stemming which is the process of reducing words to their stems or roots. A stem is the portion of a word that is left after removing its prefixes and suffixes. For example if user query is "search nearest engineering colleges" then remove "s" from colleges it remains stem that is "college".

2. SI-Indexer

The proposed system is based on the spatial inverted index. SI-index is an compressed version of I-index with compressed coordinates. Compression eliminates the defect of I-index such that SI-index consumes much less space. Compression methods are used in order to reduce size of index where each inverted index contains only IDs. To build SI-index we have used an effective method to record gaps between consecutive IDs when compared with precise ids. For example, consider a set of integers {2,3,6,8}, the gap-keeping method stores values such as {2,1,3,2} illustrate in [1]. To calculate gap between ids and coordinates of points, first it takes binary form of id and coordinates of points. Then merge binary numbers to get converted values of points based on Z-curve.

Let (x, y) be the point, then first apply $f(x)$ and $f(y)$on x and y values, where f is the function which converts decimal to binary. For example, $p_2\,(3,3)$as shown in Fig.1 that is $f(3) = 011$, $f(3) = 011$.Then let consider output of $f(x)$or $f(y)$ as set of binary bits $\{x_1 x_2 \ldots\ldots x_n\}$ where x_i is bit having value output.

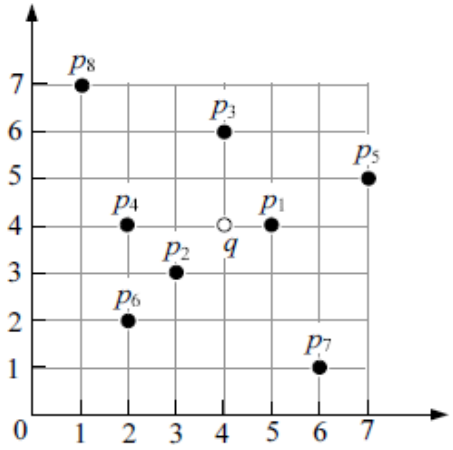

Figure 2: Shows locations of points

Next we merge $f(x)$and $f(y)$

$m(f(x), f(y))$ ..... $eq(1)$

Where, m is the merger function will do bit by bit merging of $f(x)$ and$f(y)$ output.

So if $f(x) = \{x_1 x_2 \ldots\ldots x_n\}, f(x) = \{y_1 y_2 \ldots\ldots y_n\}$

Therefore,

$m(f(x), f(y)) = \{x_1 y_1 x_2 y_2 \ldots\ldots x_n y_n\}$..... $eq(2)$

Then $d(m)$ be the function which converts bits to decimal. For example,

$f(3) = 011$

$f(3) = 011$

$m(f(3), f(3)) = 001111$

Therefore $d(m) = 15$

p_6	p_2	p_8	p_4	p_7	p_1	p_3	p_5
12	15	23	24	41	50	52	59

Fig.3 Converted values of the points in Fig.4 based on Z-curve.

The compressed spatial inverted index has a triplet to represent both ID and coordinates of each point. But gap-keeping is applied on only attribute of triplet. So it is not

simple solution. To tackle this problem we first focuses on the coordinates. Even though each point has two coordinates, we can convert that coordinates into only one attribute so that gap-keeping can applied effectively. A space filling curve (Z-curve) is constructed for efficient processing, this is called 2D Z-curve generation. Pseudo ids are used instead of real id and form 3D Z-curve. This is done for better sorting procedure. Values in Z-curve are sorted.

Let element in set $e = \{e_1, e_2, \dots, e_n\}$ integer values

Inverted index at e,

$$e = e^{th}\ value - (e-1)^{th}\ value \quad \dots\dots\dots\dots eq(3)$$

Except first element.

$$e = e^{th}\ value + (e-1)^{th}\ value \quad \dots\dots\dots\dots eq(4)$$

This gives first level gap-keeping values. Fig. 3 that contains p2, p3, p6, p8, whose Z-values are 15,52,12,23 respectively, and pseudo-ids being 1,6,0,2 respectively. After sorting Z-values automatically the pseudo ids are kept in ascending order. Gap-keeping is applied on both Z-values and pseudo-ids, it results in 12,3,8,29 and 0,1,1,4 respectively. We can capture the 4 points with 4 pairs: {(0,12),(1,3),(1,8),(4,49)}.

Two level gap-keeping method applied on values obtained in first level gap-keeping that is both values Z-values and pseudo-ids like 12,3,8,29 and 0,1,1,4 respectively. It gives result as 3,5,4,17 and 0,1,0,3 respectively. We can capture the 4 points with 4 pairs:{(0,3),(1,5),(0,4),(3,17)}. Second level gap between values is calculated using same formula that are used to calculate first level gap between values.
Let elements in set $i = \{i_1, i_2, \dots, i_n\}$ integer values.

Inverted index as i,

$$i = i^{th}\ value - (i-1)^{th}\ value \quad \dots\dots\dots\dots eq(4)$$

Except first element.

$$i = i^{th}\ value + (i-1)^{th}\ value \quad \dots\dots\dots\dots eq(5)$$

In our contribution we have built a spatial inverted index using compression method two level gap-keeping that consumes less space than single level gap-keeping and also the space cost is less when SI-index is used.

C. Searching

ID	Point name	Words
0	p0	a, b, c
1	p1	b, d, f
2	p2	d
3	p3	e, g, c
4	p4	c, d, e, f, g
5	p5	a, g
6	p6	b, f
7	p7	c, e
8	p8	d, g

Figure 4: Example of inverted index

When user enter query, then that query contains keyword and two coordinates that is (x, y). We have points (i.e two co-ordinates) and related keywords in our database which are taken from Google Map. Then search query into inverted index, it gives us points that contain query keyword. After that search for nearest one from that points. Above example shows example of inverted index that have points and related words.

D. Mobile App

User can enter a query through Mobile App.GPS location tracker can track current location information. And, finally results are stored on server.

E. Mathematical Module

Let q be the set of query keywords, these keywords are input to the system for search.

$q = \{q_1, q_2, \dots\dots q_n\}$

Let r be the dataset which is either static or dynamic, it gives preprocessing result after searching query keywords into dataset.

$r = \{r_1, r_2, \dots\dots r_2\}$

Let d_i be the distance, it is preprocessing result getting after searching query in dataset.

$d_i = \{d_1, d_2, \dots\dots d_n\}$

$d_i = search(r, q)$

Let d_{ni} be the nearest distance, which is calculated by applying k-nearest neighbor algorithm on distance d_i, which is preprocessing result of search.
Where,

$d_{ni} = kNN(d_i)$

The d_i gives distances which are result of our query but it returns number of results, form that we have to calculate minimum distance from current location so, the algorithm kNN apply on preprocessing results to calculate minimum distance from current location.

F. Algorithm

Algorithm 1 SI index Building Algorithm

Output: Semantic nearest neighbor.
Description:
1: Read three values as (id, x, y)
Where, id - id of place or word
x - Position on x-axis of place
y - Position on y-axis of place

//2D gap-keeping
2: Apply gap-keeping on x and y first as following,
a. Read x
b. Let 2D = {b1b2b3.....bn}
Convert x to binary values as,
b1= binary(x);
c. Repeat step a and b for y and create b2.
d. Merge b1and b2bit by bit and store in b3
For each bit in b1and b2,

B = b1&b2;

e. Convert B to decimal.

f. Generate Z-curve using gap-keeping.

3: Repeat step 1 and 2 for all places.

4: Generate sorted 2D Z-curve using gap-keeping.

//3D gap keeping

5: Repeat 1 and 2 for id as x and value from set 2D as y will store merge results

Let, 3D = {c1, c2.....cn}

6: Generate 3D Z-curve using gap-keeping on set 3D.

7: Apply 2-level gap-keeping on set 3D.

4. Experimental Setup

To search nearest neighbor, system requires to track current location of the user. It can be tracked by using GPS location tracker.

5. Result and Dataset

We have used "Google Map" of particular area, by selecting any location on that map it gives longitude and latitude. That longitude and latitude is saved into the database. It also includes information related to hospital, school, bank, hotel, restaurant, college etc. Fig 5 shows that we can add location information using "Google map". In Fig 6. we can add different categories like hotel from their websites. And make list.

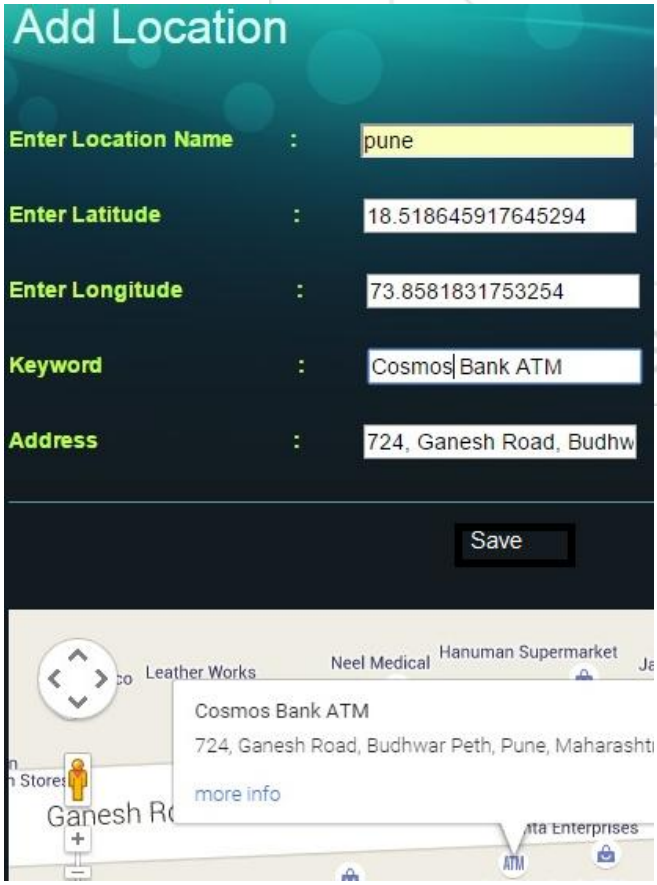

Figure 5: Add Location Information using Google Map

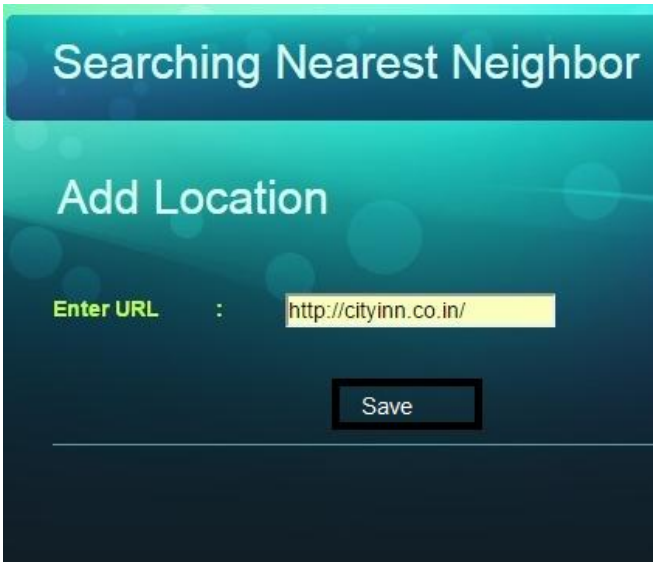

Figure 6: Add Location Information using Website

6. Conclusions

In this paper we mainly focus on spatial data mining technique. Spatial inverted Index structure is used to deal with the problem of IR2-tree. Compression of SI-index has done using Gap-keeping method. This method can't be applied on triplet. So firstly consider 2D Z-curve values and then 3D Z-curve values. And to calculate these Z-curve values there are two steps that is binary representation and merging. In this paper we used two level gap-keeping to save space cost.

7. Acknowledgement

I express great many thanks to Prof. S. S. Nandgaonkar for his great effort of supervising and leading me, to accomplish this fine work. Also to college and department staff, they were a great source of support and encouragement. To my friends and family, for their warm, kind encourages and loves. To every person gave us something too light my pathway, I thanks for believing in me.

References

[1] Yufei Tao and Cheng Sheng, "Fast Nearest Neighbor Search with Keywords," IEEE Transactions on Knowledge and Data Engineering, 2013, p1-13.

[2] D. Zhang, Y.M. Chee, A. Mondal, A.K.H. Tung, and M. Kitsuregawa, "Keyword Search in Spatial Databases: Towards Searching by Document," Proc. Int'l Conf. Data Eng. (ICDE), pp. 688-699, 2009.

[3] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma, "Hybrid index structures for location-based web search," In Proc. of Conference on Information and Knowledge Management (CIKM), pages 155–162, 2005

[4] R. Hariharan, B. Hore, C. Li, and S. Mehrotra, "Processing spatial- keyword (SK) queries in geographic information retrieval (GIR) systems," In Proc. of Scientific and Statistical Database Management (SSDBM), 2007.

[5] I. D. Felipe, V. Hristidis, and N. Rishe, "Keyword search on spatial databases," In Proc. of International Conference on Data Engineering (ICDE), pages 656–665, 2008.
[6] X. Cao, G. Cong, and C. S. Jensen, "Retrieving top-k prestige-based relevant spatial web objects," VLDB, 3(1):373–384, 2010.
[7] Y.-Y. Chen, T. Suel, and A. Markowetz, "Efficient query processing in geographic web search engines," In Proc. of ACM Management of Data (SIGMOD) , pages 277–288, 2006.
[8] X. Cao, G. Cong, C.S. Jensen, and B.C. Ooi, "Collective Spatial Keyword Querying," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 373-384, 2011.