ISSN:0975-9646

Ranking Objects by Evaluating Spatial Points through Materialized Datasets

K.Swathi[1], B.Renuka Devi[2],

M.Tech Student[1], Assoc.Professor[2]
Vignan's Lara Institute of Technology & Science

Abstract: Ranking spatial objects can be done based on their features. Every object has its own features. Based on the features, the objects could be provided with some scores likewise identifying scores for each object we could rank them. For providing scores of objects, if they were of multidimensional or high dimensional we need to use some algorithms that could provide the best objects based on their features. Likewise here we want to rank the spatial objects by using both their feature and their surrounding objects. For this purpose, here we use some dominant relationship algorithm that could give best objects that were dominant with all the features and then we will find out the best objects based on their surroundings through distance metric, by using spatial data structures and feature join algorithm through hash indexing. By this we can provide best ranked results for spatial objects.

Keywords: Skyline Points, Materialisation, M-Trees

1. INTODUCTION:

Spatial database systems manage large collections of geographic entities, which apart from spatial attributes contain nonspatial information (e.g., name, size, type, price, etc.). Here presented an interesting type of preference queries, which select the best spatial location with respect to the quality of facilities in its spatial boundaries. Ranking the spatial data can be done by using queries, A spatial preference query ranks objects based on the features in their neighborhood. For example if we want to check for a best residence i.e., for a house we will check some features like the area, price, area type or so and also we may check whether there were any hospitals, transportation facilities, educational institutes, groceries and so. Generally, if we check for any websites they will provide the list of their surroundings but wont check for them in that area, so that here we want to them check them with their latitude and longitude points by taking their distances.

Ranking spatial data depends on their features and neighbourhood data points. Here, the search results for spatial data varies with each of the sites.(for eg: Googlemaps, bingmaps, mapquest etc:) Ranking of the data is provided in different ways in Google maps scenario they have used the relevance, distance and prominence basis. So, there are two basic ways for ranking objects; Spatial ranking, which orders the objects according to their distance from a reference point and Non spatial ranking, which orders the objects based on their non-spatial values.

2. LITERATURE SURVEY:

For finding the best ranked objects among a set of objects in the existing systems, the score of each object is defined in terms of, the maximum quality for each feature in the neighborhood region and the aggregation of those qualities. A simple score instance, called the range score, binds the neighborhood region to a circular region at p with radius r shown as a circle in Fig.1a. Spatial preference query integrates these two types of ranking. These provide good results for many decision making applications.

There is no such existing solution for processing the spatial preference query. An approach, brute force is used to evaluate it to compute the scores of all objects and finds out the top ones. But this method, is to expensive for large input data sets. So there were some alternative techniques that are developed at minimizing the I/O accesses to the object and feature data sets, which were computationally efficient. These techniques are applied on spatial data structures, like R-trees and takedown upper score values for the objects indexed by them, which are used to prune the search space. Also we can use the branch-and-bound (BB), feature join (FJ) algorithm referred in[1] for processing the spatial preference query efficiently.

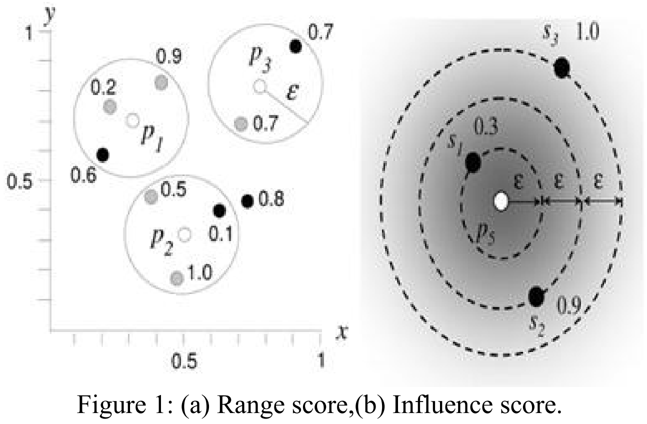


Figure 1: (a) Range score,(b) Influence score.

Ranking objects is a very important task in various applications. In relational databases, we rank tuples using an aggregate score function on their attribute values. For example, consider a property database that contains information of multiple values or multidimensional data. If we want to rank, the top 10 among the tuples, with the large size and lesser or moderate price. In this case, the score of each property is taken by the sum of two qualities, size and price. In spatial databases, ranking is done by nearest

neighbor (NN) retrieval [5]. Given a query for a property in some location, which needs to get a set of nearest objects those qualify a particular condition. These set of objects is taken to an R-tree, then apply for distance values and takedown the index in a branch-and-bound [BB] fashion to get the result.
Also, it is not so easy to use multi-dimensional indexes for getting top ones. So for such applications we need to break them in to high-dimensional spaces. Then top-k queries have to materialize attributes for combinations because they are very expensive to create & maintain. Then it could be taken as distributed database manner where every attribute will be checked in different databases and collecting all of them to be unified. So by effective merging of all the databases result could be obtained but accessing of the data will be somewhat efficient by obtaining the best result for each of the attribute.
Here the ranking process goes as follows, first review the R-tree, which are the most popular spatial access method and the NN search algorithm. Then, survey our feature-based spatial queries.
In the following sections, we propose the process for ranking the spatial data. Section 3.1 gives out the process of ranking. Section 3.2 explains the spatial access method used i.e., M-trees. Section 3.3 explains the top-k computing skyline one scan algorithm that scans the data. Section 3.4 gives the extension algorithm for our proposed system. Section 3.5 proposes the extension to the computing algorithm for ranking the data i.e., our exact procedure using hash tables and FJ algorithm.

3. RANKING SPATIAL OBJECTS:

Ranking the objects can be done by evaluating the spatial skyline points; the process for ranking will be explained in the following sections.

3.1 Process of Ranking

1. First we need to identify the places from the data.
2. Then we need to identify the neighbors and have to construct M-Trees.
3. Apply *one scan algorithm of dominant relationship* on data based on the preferred features to obtain the skyline points.
4. Create feature datasets according to their distances.
5. Applies *Skyline feature join algorithm* for the data that generates hash list according to the features with their respective locations.
6. After getting the feature data sets, the locations will be sorted based on scores.
7. Here by adding the hash indexes of each location we rank the top results.

3.2 M-Trees:

M-trees [12] are tree data structures that are similar to R-trees and B-trees. It is constructed using a metric and relies on the triangle inequality for efficient range and k-NN queries. While M-trees can perform well in many conditions, the tree can also have large overlap and there is no clear strategy on how to best avoid overlap. In addition, it can only be used for distance functions that satisfy the triangle inequality, while many advanced dissimilarity functions used in information retrieval do not satisfy this.

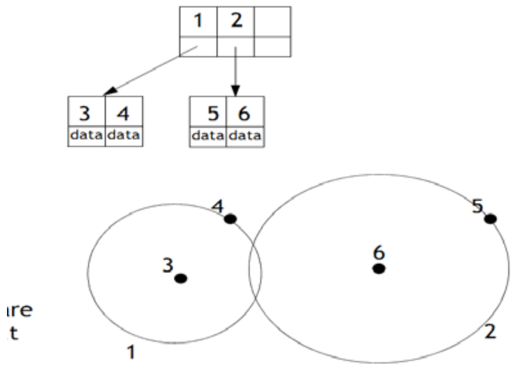


Figure 2: M-Tree

As in any Tree-based data structure, the M-Tree is composed of Nodes and Leaves. In each node there is a data object that identifies it uniquely and a pointer to a sub-tree where its children reside. Every leaf has several data objects. For each node there is a radius that defines a Ball in the desired metric space. Thus, every node and leaf residing in a particular node is at most distance from, and every node and leaf with node parent keep the distance from it.

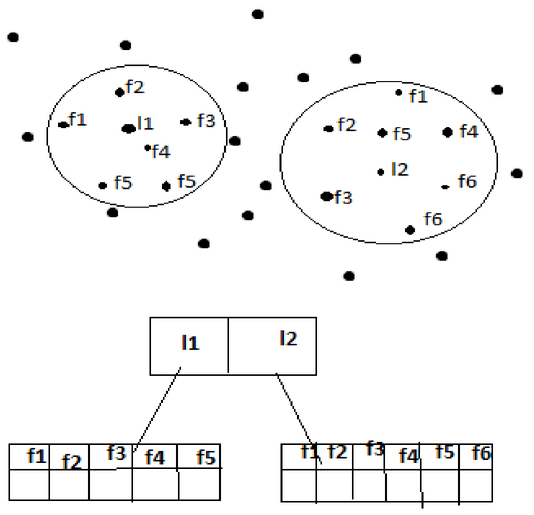


Figure 3: M-Tree For The Locations L1,L2

Materialized Data sets, materialization refers to collecting the data i.e., confined to some specific details. Suppose we need to collect only some of the features among the high dimensional data, we can materialize only those values. So whenever we are creating an M-Tree we can confine the features bounding it. So that we reduce the burden on database but we need to provide the details what data should be taken and what not. Also there were some more materialization techniques, partial or full materialization. Here materialization is used for pairing technique.
Here as proposed in the abstract, a pairing technique could be provided for the features and the objects, this acts as the technique. As the M-trees could be developed will two data

values in the pairs with the feature name and the distance as it uses the range score or neighborhood retrieval function for identifying the nearest object also saves it with its distance.
So that here in the above example l1,l2 are the two locations all the neighbors to the location are identified w.r.t M-trees and these are represented in a tree fashion as l1,l2 are the roots and the neighboring locations were taken as the leaf nodes w.r.to features. So that, the features could be identified and can be paired for further use.

3.3 One Scan Algorithm of Dominant Relationship:

Here in this step it identifies the top-k skyline points, which could be referred from reference [7] A skyline is a subset of points in the data set that are not dominated by any other points. Skyline queries, which return skyline points, are useful in many decision making applications that involve high dimensional data sets. Given a *d*-dimensional data set, a point *p* dominates another point *q* if it is better than or equal to *q* in all dimensions and better than *q* in at least one dimension. Here we apply the dominant relationship algorithms for obtaining them, the existing algorithms for computing free skylines cannot be used directly for computing *k*-dominant skyline points. Here there are three novel algorithms, namely, One-Scan algorithm, Two-Scan algorithm and Sorted Retrieval algorithm, to compute *k*-dominant skyline points. Each algorithm takes as input a *d*-dimensional data set *D* (over a set of dimensions *S*) and a parameter *k*, and outputs the set of *k*-dominant skyline points in *D*. Here we use the one scan as it is sufficient for our process.

One-Scan Algorithm

To compute *k*-dominant skyline points from an input data set *D* (over a set of dimensions *S*) is similar in spirit to the nested-loop approach in that it makes one sequential scan of the data set. The algorithm based on the following two key properties.

P1. There must exist a free skyline point in D that k-dominates p.

P2.It is possible for p not to be k-dominated by any k-dominant skyline point.

Following is the algorithm referred in [7] which provides the skyline points.

Algorithm: One-Scan Algorithm (*D*, *S*, *k*)

```
1: sort $D$ in non-ascending order of sum of point's dimension values
2: initialize set of $k$-dominant skyline points $R = \emptyset$
3: initialize set of unique non-k-dominant skyline points $T = \emptyset$
4: for every point $p \in D$ do
5:    initialize isUniqueSkyline = true
6:    for every point $p^- \in T$ do
7:       if ($p$ dominates $p^-$) then
8:          remove $p^-$ from $T$
9:       else if ($p^-$ dominates $p$) or ($p = p^-$) then
10:         isUniqueSkyline = false
11:         break out of inner for-loop
12:   if (isUniqueSkyline) then
13:      initialize isDominant = true
14:      for every point $p^- \in R$ do
15:         if ($p^-$ $k$-dominates $p$) then
16:            isDominant = false
17:         if ($p$ $k$-dominates $p^-$) then
18:            move $p^-$ from $R$ to $T$
19:      if (isDominant) then
20:         insert $p$ into $R$
21:      else
22:         insert $p$ into $T$
23: return $R$
```

So by using the skyline points R, we could get the particular locations or objects that were perfect for our spatial query so that we can directly list the objects. So after obtaining all the skyline points we list the objects with their features according to M-trees.

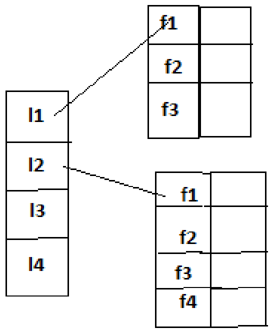


Figure 4: D, Listing the Skyline points

3.4 Algorithm for ranking the data

Here in the proposed algorithm the input we take is the Skyline points R, resulted from One scan Algorithm and Features user preferred f).

Algorithm: Ranking Data(R,f):

```
1.  Initialize RR, a set of k-dimensional values. RR=Ǿ
2.  Initialize F, a hash list F=f;
3.  Initialize D, D=R.
4.  Check all the data in D for the features in F.
5.  If(D->f1=feature) then
    F1->l1=d1
6.  for(D≠NULL)
    apply step 5.
7.  Insert Result in to F.
8.  Sort f1->(data,Score) with score.
9.  Apply step 8 for all the features in F.
10. Take D1[index]  for all the features in F.
11. Add all the indexes for D data.
12. Apply step[11] for all the data D.
13. Insert result in to RR .
14. Sort RR for ranked result.
15. Result RR.
```

The result RR provides the ranked order for the skyline points. The following section explains the Process of the algorithm.

3.5 Extending the algorithm for Ranking Objects:

Creating feature data sets refers to creating object sets for each feature. For this we use the Feature Join Algorithm. Then we need to place them in a hash table.

Feature Join Algorithm:

This is also a method for evaluating preference query to perform a multi-way spatial join on the features f1, f2, f3….fn to obtain combination of feature points which can be in the neighborhood of objects from R. Spatial regions which correspond to combinations of high scores are then examined, in order to find data objects in R having the corresponding feature combination in their neighborhood.

Here we first take the combinations i.e., location for each feature. Then we need to place all the location for that feature f1{ l1,l2,l3…ln},f2{l1,l2,l3…ln},…fn{l1,l2,l3…ln}. Here after creating datasets, we need to create a hash table for all the features with user preference as a hash function i.e., priority of user. Then the buckets for each of the hash entry will be the locations which are placed according to their distance in the minimum distance order. This could be represented in the following,

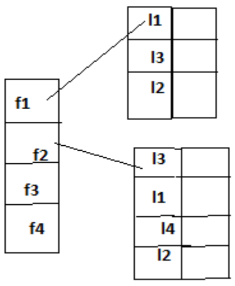


Figure 5: H, Listing the features in hash table according to Join Feature Algorithm

So that all the features f1, f2, f3, f4,.. fn are listed in the hash Table. The locations for the feature f1 are entered in to their buckets, which are entered based on their distance, but here the distance is confined to some range as we selected the neighbors using Range Query in M-Trees.

So that, the locations were again sorted according to their feature scores that were given by some rating mechanism which were not discussed here.

Ranking the objects:

Here ranking the objects can be done by adding the indexes of the location in all the feature buckets. So that by collecting all the indexes added for each location for every feature, we get the position for each location. So that by sorting all the results we could rank the locations. So according to the ranking order the locations will be ranked.

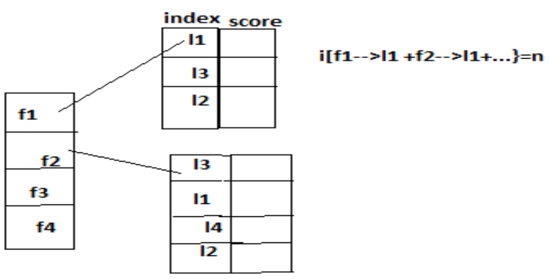


Figure6: Ranking the locations.

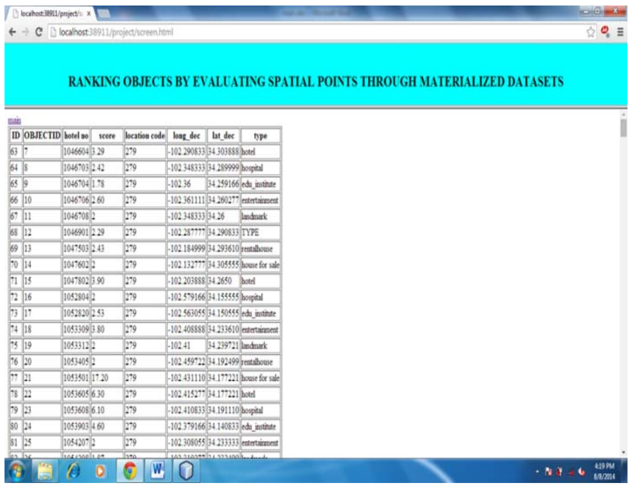
The ranking of the locations can be described as follows: F1-->L1[i], f2-->L1[i], f3-->l1[i],……fn-->l1[i]; Here i ,refers to indexes of that location.

Here we need to take all the indexes and add them as follows:

index , i[f1-->l1]+i[f2-->l1]+ i[f3-->l1]+ i[f4-->l1]+…..i[fn→l1]=n1., like wise adding all the indexes we could get their position scores. Take the result in to RR. Sort the data in RR, according to their added index scores. So that we could say which one's position score is less it could be ranked first as best location. Here, as we use the skyline points there is no problem in checking all the features. As all the locations or the points that were taken will possess all the locations or all the features that are needed.

4. Results:

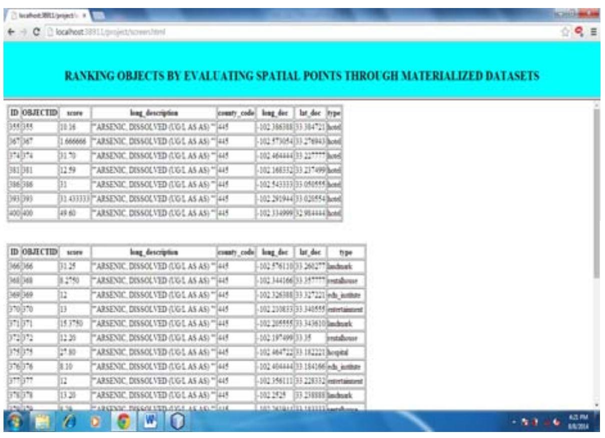
The proposed algorithm is applied on a database which had more than 500 entries with their latitude and longitude points. Which was the data taken from Texas, Lubbock wells information as we mostly needed the latitude and longitudinal data as we need to check on large data. So, that to check the time it takes. We have made some changes for the data like hotels and some as the features that we need. First displays the full data



RANKING OBJECTS BY EVALUATING SPATIAL POINTS THROUGH MATERIALIZED DATASETS

ID	OBJECTID	hotel no	score	location code	long_dec	lat_dec	type
63	7	1046604	3.29	279	-102.290833	34.303888	hotel
64	8	1046703	2.42	279	-102.348333	34.289999	hospital
65	9	1046704	1.78	279	-102.36	34.259166	edu_institute
66	10	1046706	2.60	279	-102.361111	34.260277	entertainment
67	11	1046708	2	279	-102.348333	34.26	landmark
68	12	1046901	2.29	279	-102.287777	34.290833	TYPE
69	13	1047503	2.43	279	-102.184999	34.293610	rentalhouse
70	14	1047602	2	279	-102.132777	34.305555	house for sale
71	15	1047802	3.90	279	-102.203888	34.2650	hotel
72	16	1052804	2	279	-102.579166	34.155555	hospital
73	17	1052820	2.53	279	-102.563055	34.150555	edu_institute
74	18	1053309	3.80	279	-102.408888	34.233610	entertainment
75	19	1053312	2	279	-102.41	34.239721	landmark
76	20	1053405	2	279	-102.459722	34.192499	rentalhouse
77	21	1053501	17.20	279	-102.431110	34.177221	house for sale
78	22	1053605	6.30	279	-102.415277	34.177221	hotel
79	23	1053608	6.10	279	-102.410833	34.191110	hospital
80	24	1053903	4.60	279	-102.379166	34.140833	edu_institute
81	25	1054207	2	279	-102.308055	34.233333	entertainment

The following displays the hotels with some score specified above some value and their neighbors to a particular location.

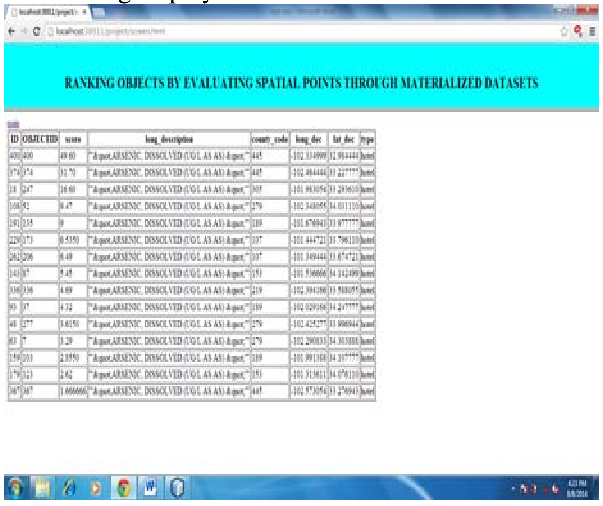


RANKING OBJECTS BY EVALUATING SPATIAL POINTS THROUGH MATERIALIZED DATASETS

ID	OBJECTID	score	long_description	county_code	long_dec	lat_dec	type
355	355	10.16	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.386388	33.384721	hotel
367	367	1.666666	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.573054	33.276943	hotel
374	374	31.70	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.464444	33.227777	hotel
381	381	12.59	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.168332	33.237499	hotel
386	386	31	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.543333	33.050555	hotel
393	393	31.433333	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.291944	33.020554	hotel
400	400	49.60	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.334999	32.984444	hotel

ID	OBJECTID	score	long_description	county_code	long_dec	lat_dec	type
366	366	31.25	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.576110	33.260277	landmark
368	368	8.2750	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.344166	33.357777	rentalhouse
369	369	12	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.326388	33.327221	edu_institute
370	370	13	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.230833	33.340555	entertainment
371	371	15.3750	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.205555	33.343610	landmark
372	372	12.20	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.197499	33.35	rentalhouse
375	375	27.80	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.464722	33.182221	hospital
376	376	8.10	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.404444	33.184166	edu_institute
377	377	12	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.356111	33.228332	entertainment
378	378	13.20	""ARSENIC, DISSOLVED (UG/L AS AS) ""	445	-102.2525	33.238888	landmark

The following displays the best hotels in that area ranked.

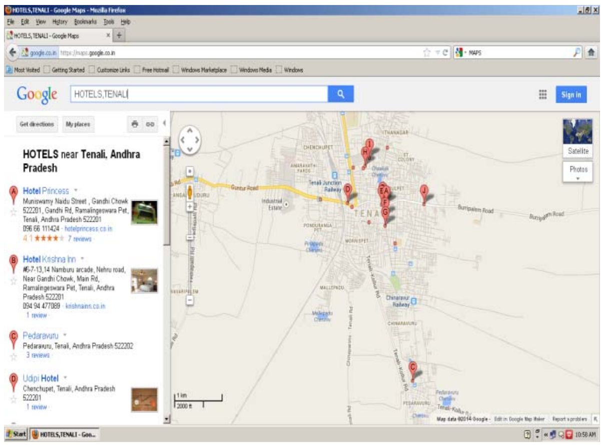


RANKING OBJECTS BY EVALUATING SPATIAL POINTS THROUGH MATERIALIZED DATASETS

ID	OBJECTID	score	long_description	county_code	long_dec	lat_dec	type
400	400	49.60	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	445	-102.334999	32.984444	hotel
374	374	32.70	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	445	-102.464444	33.227777	hotel
18	247	16.60	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	305	-101.983056	33.293610	hotel
108	52	9.47	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	279	-102.348055	34.031110	hotel
191	135	9	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	119	-101.676943	33.077777	hotel
229	173	8.5350	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	107	-101.444721	33.796110	hotel
262	206	6.48	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	107	-101.349444	33.674721	hotel
143	87	5.45	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	153	-101.516666	34.142499	hotel
336	336	4.69	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	219	-102.394166	33.588055	hotel
93	37	4.32	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	189	-102.029166	34.247777	hotel
48	277	1.6150	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	279	-102.425277	33.996944	hotel
63	7	1.29	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	279	-102.290833	34.303888	hotel
159	103	2.8550	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	189	-101.991388	34.397777	hotel
179	123	2.62	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	153	-101.313611	34.076110	hotel
367	367	1.666666	"&quot;ARSENIC, DISSOLVED (UG/L AS AS) &quot;"	445	-102.573056	33.276943	hotel

To check the accuracy we have applied the proposed algorithm, on Google maps ranked data. So far the results were merely accurate.

The following display the result from Google Maps for searching hotels in Tenali. As Google uses the relevance, distance and prominence criteria's.



HOTELS near Tenali, Andhra Pradesh

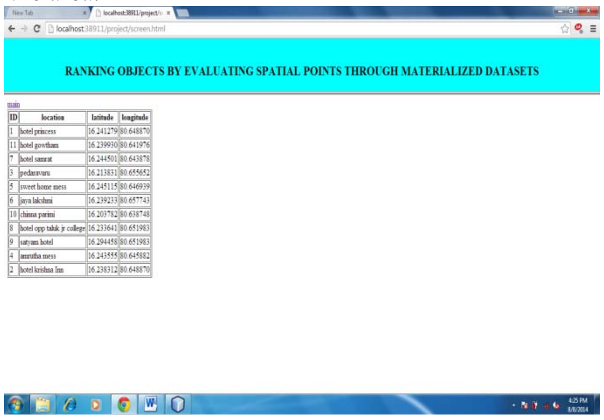
Hotel Princess
Munuswamy Naidu Street, Gandhi Chowk 522201, Gandhi Rd, Ramalingeswara Pet, Tenali, Andhra Pradesh 522201
096 66 111424 · hotelprincess.co.in
4.1 ★★★★ 7 reviews

Hotel Krishna Inn
#5-7-13,14 Namburu arcade, Nehru road, Near Gandhi Chowk, Main Rd, Ramalingeswara Pet, Tenali, Andhra Pradesh 522201
094 94 477089 · krishnainn.co.in
1 review

Pedaravuru
Pedaravuru, Tenali, Andhra Pradesh 522202
3 reviews

Udipi Hotel
Chenchupet, Tenali, Andhra Pradesh 522201
1 review

The following displays the ranked results from the data gathered through Google maps. We had taken the result from Google and worked out with the features like bus stand and railway station, nearer to the hotels w.r.to Latitude and Longitude points and also the scores of hotels in the area.

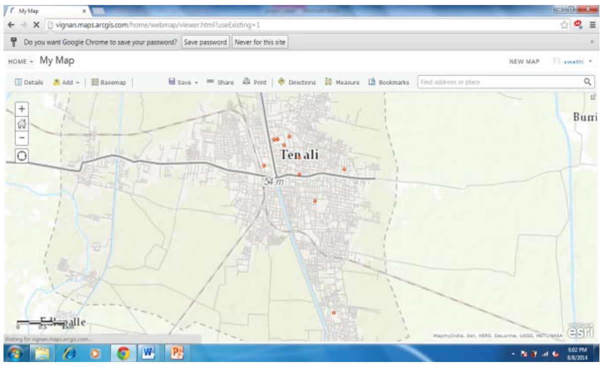


RANKING OBJECTS BY EVALUATING SPATIAL POINTS THROUGH MATERIALIZED DATASETS

main

ID	location	latitude	longitude
1	hotel princess	16.241279	80.648870
11	hotel gowtham	16.239930	80.641976
7	hotel samrat	16.244501	80.643878
3	pedaravuru	16.213831	80.655652
5	sweet home mess	16.245115	80.646939
6	jaya lakshmi	16.239233	80.657743
10	chinna parimi	16.203782	80.638748
8	hotel opp taluk jr college	16.233641	80.651983
9	satyam hotel	16.294458	80.651983
4	amrutha mess	16.243555	80.645882
2	hotel krishna Inn	16.238312	80.648870

The following provides the resultant for visualization on maps using the ARCGIS online tool.



5. CONCLUSION:

By materializing all the dimensions one by one here in the computing algorithms, the skyline points could be accurate. Also the extension algorithm provides the most useful skyline points that makes useful to make the right decision. Here in the paper we have given the ranking extension to the skyline point retrieval algorithm by computing the skyline points using Feature join algorithm and Hash tables by materializing using the M-Trees. By using the entire proposed algorithm the results could be as they check all the surrounding places to it by using M-Trees so that to make right decision.

6. FUTURE ENHANCEMENT:

The extension to this paper can be done by identifying the area of the location for any purpose suppose we need an area of 5 acres of residential area or so, by using the Pattern recognition techniques like chain coding. So that we can identify an area in a map and those could be ranked using the ranking techniques.

REFERENCES:

1. Man Lung Yiu; Hua Lu; Nikos Mamoulis; Vaitis, M., "Ranking Spatial Data by Quality Preferences," Knowledge and Data Engineering, IEEE Transactions on , vol.23, no.3, pp.433,446, March 2011
2. Evaluating Top-k Skyline Queries over Relational Databases, Database and Expert Systems Applications,Lecture Notes in Computer Science Volume 4653, 2007, pp 254-263
3. Lijiang Chen; Bin Cui; Hua Lu, "Constrained Skyline Query Processing against Distributed Data Sites," Knowledge and Data Engineering, IEEE Transactions on , vol.23, no.2, pp.204,217, Feb. 2011
4. Yong Sung Kim; HaRim Jung; Min Kyung Sung; Yon Dohn Chung, "On processing scored k-dominant skyline queries," Electrical and Control Engineering (ICECE), 2011 International Conference on , vol., no., pp.4834,4837, 16-18 Sept. 2011
5. Xin Lin; Jianliang Xu; Haibo Hu, "Range-Based Skyline Queries in Mobile Environments," Knowledge and Data Engineering, IEEE Transactions on , vol.25, no.4, pp.835,849, April 2013
6. Skyline computation, by jian wen, in CSG399:http://www.ccs.neu.edu/home/jarodwen/materials/skyline_pre.pdf
7. Extract interesting skyline points in High Dimension, Database Systems for Advanced Applications,Lecture Notes in Computer Science; Volume 5982, 2010, pp 94-108
8. Fung, G.P.C., Lu, W., Du, X.: Dominant and k nearest probabilistic skylines. In: Proceedings of the 14th International Conference on Database Systems for Advanced Applications, DASFAA 2009
9. Zhang, S., Mamoulis, N., Cheung, D.W.: Scalable skyline

computation using object-based space partitioning. In: Proc. of the 35th SIGMOD International Conference on Management of Data(SIGMOD 2009)

10. Efficient Processing of Top-k Dominating Queries on Multi-Dimensional Data, VLDB ‘07, September 23-28, 2007, Vienna, Austria.Copyright 2007 VLDB Endowment, ACM 978-1-59593-649-3/07/09
11. M.L. Yiu, X. Dai, N. Mamoulis and M. Vaitis, "Top-k Spatial Preference Queries," Proc. IEEE Int',l Conf. Data Eng. (ICDE)
12. Guttman, "R-Trees: A Dynamic Index Structure for Spatial Searching," Proc. ACM SIGMOD
13. Junyi Chai; Liu, J.N.K.; Man Lung Yiu; Hongwei Wang; Anming Li "A Novel Dynamic Skyline Operation for Multicriteria Decision Support", System Sciences (HICSS), 2013 46th Hawaii International Conference on, On page(s): 1183 – 1192
14. Beomseok Nam; Sussman, A. "DiST: fully decentralized indexing for querying distributed multidimensional datasets", Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International
15. Beomseok Nam; Sussman, A. "Spatial indexing of distributed multidimensional datasets", Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International Symposium on, On page(s): 743 - 750 Vol. 2 Volume: 2, 9-12 May 2005