Representation of spatial proximity for various spatial data using Geometric methods

M. Venkatesan, Vivekanand Khyade

Abstract— Co-location pattern is the subsets of Boolean spatial features whose instances are often located in close geographic proximity. Representation of proximity in geospatial data using conventional data models is not consistent. Neighbourhood is a major challenge and key part of spatial co-location pattern mining. Neighbourhood may be defined using topological relationship, metric relationship and combination of both. In most of the co-location mining, the neighbourhood was defined by user. In this paper, we are proposing Delaunay triangulation approach to model the neighbourhood between the objects. Delaunay triangulation is a structure representing the neighbourhood of objects in a succinct and unambiguous manner [7]. This approach eliminates the parameters from the user to define the neighbourhood of objects and avoid multiple test and trail repetitions in the process of mining. A Delaunay triangulation based co-location mining algorithm is developed to mine co-location patterns from complex spatial data.

Index Terms— Voronoi diagrams, Dynamic Delaunay triangulation, Spatial Data Clustering.

1 INTRODUCTION

AS per the requirement of fast growing world & the ever increasing applications the modelling & simulation of spatial processes is widely used. The applications such as: water resource management, forest fire monitoring, terrestrial survey & many more such applications where we use it. So in case of representation of proximity in geospatial data using the old data models that we are using for the time is not so consistent. So here the concept of Delaunay triangulation arises to find the neighborhood between the objects. In spatial co-location pattern mining neighborhood is a major challenge & key part. The neighborhood can be define by many relationship such as the distance between that two objects, the estimated time taken to reach that object or combination of both can be used. In Delaunay triangulation we are avoiding the user inputs so that user needs not to worry about the relationship between the objects. This approach even avoids multiple test and trail repetitions in the process of mining. As the spatial data is very much complex compared to others it's mining can be easily done through Delaunay triangulation.

As we know Data mining is a process to extract implicit, nontrivial, previously unknown and potentially useful information (such as knowledge rules, constraints, regularities) from data in huge databases. As the enormous growth of a firm the data related with it is also increasing which leads to large databases. Then successively there is a need to find the perfect tool that will transform the raw data & will represent it in the form of useful information & knowledge accurately & in timely manner. Spatial data mining as a subfield of data mining refers to the extraction from spatial databases of implicit knowledge, spatial relations or significant features or patterns that are not explicitly stored in spatial databases. The large amount of spatial data obtained from satellite & the geographic information system (GIS) should be explored in detail. The difficulties that we face in spatial data mining arise because of the following issues. 1) The classical data mining is designed to process numbers and categories & spatial data is more complex and includes ex-tended objects such as points, lines and polygons. 2) The classical data mining works with explicit inputs, whereas, spatial predicates and attributes are often implicit. Finally, classical data mining treats each input independently of other inputs, while spatial patterns often exhibit continuity and high autocorrelation among nearby features. In this paper we presents simple and effective algorithm using dynamic Voronoi diagrams and Delaunay triangulations which may be useful for a wide variety of applications in clustering spatial datasets. We have used Voronoi diagram which makes subdivision of the total area under consideration into regions. Then each point on the plane is assigned to each region individually. This planer distribution of points of the surface into separate region is called as voronoi diagram [3].

2 PROPOSED METHOD

2.1 Voronoi Diagram:

Let S = {K_1, ... ,Kn} be a set of n points in the plane.

Definition of Voronoi diagram: For any subset S ⊆A of parts we define the Voronoi region,V(A), for A to be the set of all points K ε R^2 such that A is the set of closest sites to K. There are 2^n different subsets of S, but only a very few of them have a nonempty Voronoi region, V (A).

In fact, if S has no four cocircular points, then V (A) is empty for any A having |A| ≥ 4. For each Singleton set, A = {K_i}, V (A) = V ({K_i}) is nonempty and is often called the Voronoi cell associated with site K_i. We usually abuse notation slightly and let V (K_i) = V ({K_i}). By our definition, V (K_i) is always an open, which defnes V (K_i) to be a closed set that includes its boundary. If A consists of two points, A = {K_i, K_j} and V (A) is nonempty, then V (A) = V ({K_i, K_j}) is called a Voronoi edge; it will be a subset of the bisector, b(K_i, K_j), between K_i and K_j . Voronoi edges are open line segments (they do not include their end points). If A consists of 3 or more points, then V (A) will be a singlePoint, we call such points Voronoi vertices. The Voronoi diagram for S, denoted V(S), is defined to be the partition of R^2 into the regions V (A), for all S ⊆A. The Voronoi diagram is a polygonal subdivision, consisting of vertices, edges, and (convex) polygonal faces (cells) [11].

1) Voronoi Diagram using divide & conquer method:

Experts have presented the deterministic algorithm for constructing the Voronoi diagram in the given N dimension plane which is optimal it is of worst case sense. The proposed algorithm was useful in theoretical view just because it was the first algorithm & it uses the divide-and-conquer method [6]. Divide and Conquer is very primitive technique for designing efficient algorithms. In this method, the problem which we have to solve is divided recursively into different-different modules & then the simpler problems are solved individually, and the solution of that given problem is then recoverd by merging all separate smaller problems. In the divide-and-conquer approach the set of sit is split up by a dividing line into parts P_L and P_R of about same sizes. Then, the Voronoi diagram, Vor(P_L), of subset P_L and Voronoi diagram, Vor(P_R), of subset $P_R$are computed recursively[2].

1) Voronoi Diagram:

INPUT:The number n > 3 of sets and the list p=(p1,p2,…..,p_n) such that in non decreasing order with respect to Major-axis.

OUTPUT: VORONOI_DAIGRAM.

ALGORITHM: VoronoiDia(P)

1. Let 'k' be the integral part of n/2 and Partition P into two part's.

i.e. P_L =(P_1,P_2,…..,P_k) and P_R=(P_{k+1},P_{k+2},…..,P_n).

2. Create the Voronoi _Diagram i.e. Vor(P_R) such that P_R recursively.
3. Create the Voronoi _Diagram i.e. Vor(P_L) such that $P_L$recursively.
4. Merge Vor(P_L) and Vor(P_R) into P {i.e. P = P_L + P_R}. by MERGING ALGORITHM.
5. Return Vor(P) i.e. VORONOI_DAIGRAM.

The main thing in the algorithm is to find the line which splits the diagram into 2 halves and merging the two halves two get original two parts. So if we consider the splitting and merging time of that two Voronoi diagram as O(n) and the total running time is O(n log n)[5] then the recurrence relation can be formed as T(n) = 2T (n/2) + O(n).

1) Merging Voronoi Diagrams:

This step involves computation of the perpendicular bisector s of parts P_L and P_Ri.e.,B(P_L , P_R), of all Voronoi edges of Vor(P) that distinguish the parts in P_L from regions of parts in P_R. The idea of collecting two diagram merging based on the fact that the edges of B(P_L , P_R) form a single y-monotone polygonal chain. The MERGE VORONOI algorithm runs in O(n) time.

INPUT: Voronoi diagrams Vor(P_L) and Vor(P_R) of parts $P_L$andP_R .

OUTPUT: Voronoi diagrams for set P = P_L + P_R

ALGORITHM: MERGE_VORONOI ( )

1. Generate the Convex hull of P_L and P_R.
2. Find the lower common support line L(P_L , P_R) by Algorithm Lower Common Support i.e. Algo(LCS)
3. w_o ← the extreme point at infinity downward on perpendicular bisector of sites p_l belongs to $P_L$and p_R belongs to P_R . i.e., B(P_L , P_R). i ← 0
4. While L(P_L , P_R) is not the upper support Repeat
   4.1 i←i+1
   4.2 Find the point of intersection of B(P_L, P_R) with the boundary of V(p_L), say a_L .
   4.3 Find the point of intersecton of B(P_L, P_R) with the boundary of V(p_R), say a_R .
   4.4 IF y value of a_L is smaller that y value of a_R.
   THEN
   w_i ←a_L
   p_L ← parts on the other part of the Voronoi edge containing a_L .
   ELSE
   w_i←a_R
   p_R ← parts on the other part of the Voronoi edge containing a_R.
5. m ← i. W_{m+1} ← the point at infinity upward on the perpendicular bisector of p_l belongs to $P_L$and p_R belongs toP_R . i.e., B(P_L , P_R).
6. Add the polygonal line (w0w1 , w1w2 , . . . , w_{m}w_{m+1}), and delete from Vor_L the part to the right of the polygonal line and delete from Vor R from the part to the left polygonal line.Return the resultant Voronoi diagram.

2) Support line Computation:

We will get the common support line from MERGE VORONOI algorithm as shown in algorithm LCS. This algorithm has time complexity of O(n). There is an O(log n) algorithm for finding the common support it will not create any substantial effect on the overall algorithm so we will ignore it.

Algorithm for finding LOWER COMMON SUPPORT:

INPUT: Two Convex polygon CP_L and CP_R such that CP_L is completely left of CP_R .

OUTPUT: A pair consisting of vertex u belongs to CP_L and v belongs to CP_R such thatL(u, v) forms the lower common support of polygons CP_L and CP_R.

ALGORITHM LCS ( ):

1. Find the vertex u belongs to CP_L with the largest x coordinate and the v belongs to CP_R with the smallest x coordinate.
2. Repeat sub steps 2.1 and 2.2 Alternately.
   2.1 While vertex next[x] is lower than L(x, y) Repeat
   x←next[x].
   2.2 While vertex next[y] is lower than L(x, y) Repeat
   y ← next[y].
3. Return L(x, y).

In this paper, we are formulating a voronoi diagram from which the neighbourhood locations can be extracted. The voronoi diagram consists of locations along with its neighbouring points. A location has two attributes (object_name, instance_name). Here,instance_name itself is the location.'Locations' refers to the main representative point with the help of which voronoi diagram has been drawn. 'Point' refers to the sub-points which are there in voronoi poly-

gons.

Rule for Generating Neighbourhood Locations:-

1.Find all the adjacent polygons.The locations represents the polygons are taken as neighbourhoods. At this step, we eliminate the locations which belong to the same object. For eg. If (a1 ,a2)is neighbourhood pair, then we must eliminate it because they belong to the same object A.

2. Don't allow any repetition of pair i.e. (a1,b2) and (b2,a1) is not allowed. One of them is eliminated.

3. Find neighbourhoods according to the above given rules. This is the final set of neighbourhoods we get from voronoi diagram.

2.2 Delaunay Triangulation

A triangulation of S can be said as a Delaunay triangulation, denoted by Del(S), if there does not exist any point inside the triangles formed from S. A triangle is called the Delaunay triangle if its circumcircle is empty of nodes of S inside. It is well known that the Delaunay triangulation Del (S) is a planar Euclidean graph K (S)[1].

Vor (P) is Voronoi dia. Created by set of n distinct pts. Such that $P=\{P_1,\ldots,P_n\}$ C R^2 $(3\leq n<\infty)$ it satisfy the non-co linearity.[8] $R=\{r_1,\ldots,r_{nv}\}$ be set of vertices of voronoi and $y_{j1},\ldots,y_{jkt}$ be location vertex of the created pts. whosevoronoi polygons share same vertex.

$S_j=\{y \mid y=\sum_{t=0}^{kj} \sigma_t\, y_{jt}$; Where $\sum_{t=1}^{kj} \sigma_t=1;\ \sigma_t\geq 0;\ t \in J_{k1}\}$ by using above equation we get

$D=\{S_1,\ldots,S_{nv}\}$.

If $k_j=3$ for all $j \in J_{nv}$;

We call the set D DELAUNAY TRANGULATION

Now, we will analyze the neighbourhoods which we can get from Delaunay triangles. The most important factor in Delaunay diagram is that it is a planar graph.We can get triangles of locations.

These triangles help us in mining information in more effective way [9].

Rules for generating Triangles:-

1.The locations of the same triangle are taken as neighbourhoods. At this step, we eliminate the locations which belong to the same object. For eg. If (a1 ,a2, a3)is neighbourhood triangle, then we must eliminate it because they belong to the same object A. But (a1,a2,b2) is allowed.

2. Don't allow repeatition of triangular neighbourhoods.Foreg. Triangle (a1,a2,b2) and (b2,a1,a2) are not allowed.One of them must be eliminated.

- *Author name is currently pursuing masters degree program in electric power engineering in University, Country, PH-01123456789. E-mail: author_name@mail.com*
- *Co-Author name is currently pursuing masters degree program in electric power engineering in University, Country, PH-01123456789. E-mail: author_name@mail.com*

(*This information is optional; change it according to your need.)*

The following is the algorithm for Delaunay Diagram.

Algorithm:-

1. Take number of points from user in N.
2. for i -> 1 to N
   - 2.1.Draw triangle between i and each point.
   - 2.2. Check for intersection between lines.
   - 2.3. Take all intersections in intersect[].
   - 2.4. For j -> 1 to (size of intersect[]).
     - 2.4.1.Remove the respective triangles.
   - 2.5. End of Loop
3. End of loop

2.3 Apriori Algorithm For Finding Frequently Co-located Points

The **min-max Apriori algorithm** is as follows:-

1. Take all the transactions in trans[][3].
2. Store it in item_count [].Call it C1.
3. Calculate the support for each item by the formula given below.

Support: The % of relevant data transactions for which the pattern is true.

For ex.-Support (Laptop -> pen drive) =

$$\frac{\text{No of transactions containing both laptop \& pendrive}}{\text{No of total transactions}}$$

4. Now ,find the minimum support from item_count[] by formula: -maximum_count/2.
5. Eliminate the items which has mininimum support and store all other items in prune_item_count[]. Call it as L1.
6. Take combination of length 2 from prune_item_count[] without repeatition .
7. Calaulate the count of each distinct item and store it in item_2_count[].Call it as C2.
8. Calculate the support for each item by the above given below.
9. Now ,find the minimum support from item_2_count[]:- maximum_count/2.
10. Eliminate the items which has mininimum support and store all other items in prune_item_2_count[]. Call it as L2.
11. Generate Rules from L2.
12. Calculate confidence for each rule by the following formula.

Confidence: The measure of surety or certainty associated with each pattern that we have derived.

Confidence (Laptop -> pen drive) =

$$\frac{\text{No of transactions containing both laptop \& pendrive}}{\text{No of transactions containing laptop}}$$

13. Now extract the rules which have maximum confidence.
14. Exit.

3 PERFORMANCE ANALYSIS

3.1 Comparison between Voronoi and Delaunay diagram:-

We are comparing the information given to us by both the diagrams i.e. Voronoi and Delaunay diagram in the form of neighbourhood pairs. The neighbourhood pairs we get after applying min-max apriory algorithm are less in number as compared to voronoi diagram neighbourhoodssignificantly.It gives us more precise information.For decision making, this precise information is important.

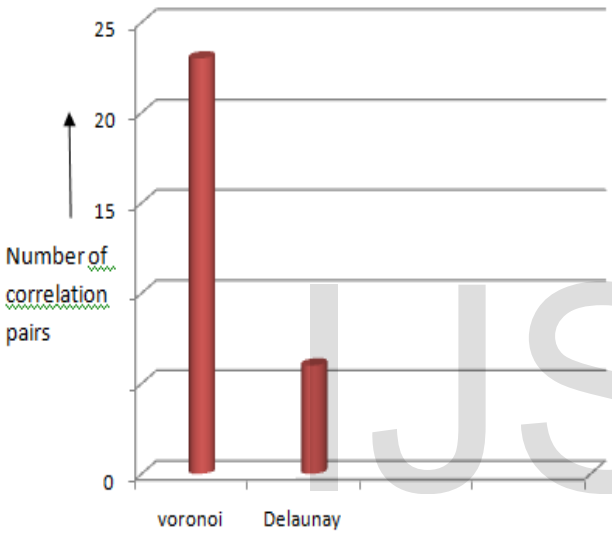

Fig 4.1 Graph for Graph plotting number of correlation pairs given by Voronoi and Delaunay diagram.

In fig 4.1, we can see that for the same set of given points , the correlation pairs given by the delaunay diagram are much informative than given by voronoi diagram. The correlation pairs given by the voronoi diagram are 23 in number. If a decision is to betaken to select one pair there is more confusing situation because the output shows large number of correlation pairs [10].In correlation pairs after applying min- max Apriory algorithm on the triangular neighbourhoods of Delaunay diagram,there is more accurate information. As the number of pairs are less i.e.6pairs , the decision making process can be more accurate.

4.CONCLUSION AND FUTURE WORK

For finding spatial proximity between locations,Delaunay diagram gives better results in terms of accuracy and precise information.It can be preferred over voronoi if we want accurate and precise information rather than bulk information.

In future, the temporal correlation between objects can be found out if we include the time stamp with each instance of the object. We are developing a module for including time stamp as a parameter with each instance which will give correlation according to the time constraints we specify.

REFERENCES

[1] F. H.Razafindrazaka,Delaunay Triangulation Algorithm and Application to Terrain Generation,22 May 2009.

[2] D. Hoey and M. I. Shamos, *Closest-point problems*, IEEE Sympos. Foind. Computer. Science,1975.

[3] Mark De Berg and Marc van Kreveld, *Computational Geometry*,Third Edition.

[4] Ashish Jain, *Apriori algorithm implementation*,Januay29,2009.

[5] Bernard Chazelle, Monique Teillaud, *Splitting a Delaunay triangulation in linear time*, INRIA.

[6] P.Cignoni, R. Scopigno, *DeWall:A Fast Divide & Conquer*, I.E.I.

[7] Siu-Wing Cheng, *Voronoi& Delaunay diagrams*, The Hong Kong university of science & technology.

[8] Joe Mitchell, *Notes on voronoi&delauay diagrams*, AMS.

[9] Dongquan Liu, Gleb V. Nosovskiy, *Effective clustering & boundary detection algorithm based on Delaunay triangulation*, ScienceDirect.

[10] Christopher M. Gold, *Problems with handling spatial data- The voronoi approach*, CISM JOURNAL.

[11] Atsuyuki Okabe, Barry Boots, Spatial tessellations: Concepts & application of Voronoi diagrams, Second edition.