Progressive Spatial Join for Polygon Data Stream

Oje Kwon
Department of Computer Science and Engineering
Pusan National University
Pusan, South Korea
kwonoje@pnu.edu

Ki-Joune Li
Department of Computer Science and Engineering
Pusan National University
Pusan, South Korea
lik@pnu.edu

ABSTRACT

Handling spatial data stream is a crucial requirement in data stream management systems (DSMSs). Especially, spatial join processing for non-point objects in DSMS is more complicated and expensive than point objects. Two critical issues arise in processing spatial join on non-point objects in DSMS. First, the size of a non-point object is larger than a point and it decreases the storage utilization of sliding windows in DSMS. It results in a low accuracy of spatial join processing. Second, the processing cost for spatial join on non-point spatial data stream increases due to the complexity of non-point objects. In this paper, we propose a method for spatial join processing on spatial data stream of polygon objects in DSMS. The key idea of our method is to represent a polygon with multiple levels of detail (LODs) and process spatial join in progressive way from low to high LOD. This strategy not only improves the storage utilization by filtering out non-overlapping polygons with low LOD, but also reduces the processing cost by simpler polygons of low LOD. Intensive experiments were carried out to analyze the performance of our methods with different parameters.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Algorithms, Management

Keywords

Spatial join, Polygon data stream, Progressive transfer, Multiple LODs, DSMS

1. INTRODUCTION

Although spatial data streams may contain non-point objects such as polygons in real applications, only point objects have been considered in spatial data stream management systems (DSMSs). Suppose that an oil spill happens at a sea and its consequences become disastrous in areas where the sea water temperature is higher than 30°C. In this case, we need to compute spatial joins of these dynamic areas, which are often given as polygon streams. However, the polygons are often complex and the data size becomes large.

It leads to two serious problems in handling spatial join on polygon data streams in DSMS. First, it may significantly degrade the accuracy of spatial join on data stream due to the limited size of sliding window. For example, if a polygon data has about 100 vertices, we can only store 13 polygons in a sliding window of 16K bytes. We cannot expect accurate results of spatial join with only 13 objects. Second, the cost of spatial join processing is quite expensive due to the geometry computation of polygons. We may fail to respect the real-time constraint of DSMS due to the high processing cost. However, most of DSMSs have been handling point[3] or rectangle[2] objects for spatial joins and these issues have been ignored in precedent researches.

In order to solve these problems, we propose a new method for spatial join processing on polygon data streams. The key idea of our method is to represent polygon objects in multiple levels of detail (LODs) and transfer them to DSMS from low to high LOD in a progressive way. The spatial join is accordingly processed with multiple LODs from simplified polygon to complex polygon. If a polygon of low LOD in the sliding window does not overlap with any other polygons of sliding window, then we do not keep it in sliding window. As a result, we increase the accuracy of spatial join by storing more polygons and reduce the CPU processing time with simplified polygons. In order to validate our approaches, we carried out intensive experiments with real and synthetic data sets and measured the performance of spatial join.

This paper is organized as follows; the concept of progressive transfer of polygon with multiple LODs and a spatial join algorithm based on the progressive transfer technique will be proposed in section 2 and 3, respectively. In section 4, we will show the results of experiments with real and synthetic data to analyze the performance of progressive spatial join method. We conclude the paper in section 5.

2. MULTIPLE LEVELS OF DETAIL FOR POLYGON DATA STREAM

One of key concepts of our approach is to represent a polygon in a progressive way with multiple LODs. In this section, we introduce the progressive representation of polygon and discuss the implementation issues.

2.1 Progressive representation of polygon

The progressive representation of a polygon with multiple LODs is defined as follows;

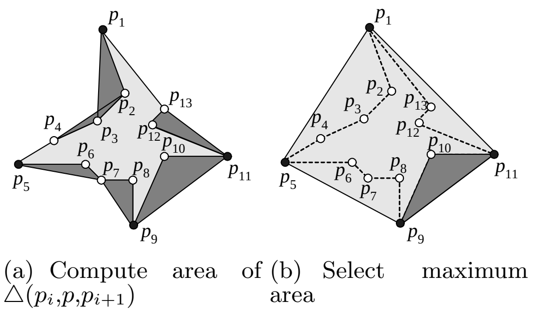
Definition Progressive polygon representation

Given a polygon A, the polygon is called the k-th ($k \in \{1,...,N\text{-}1\}$) LOD polygon of A, $\text{LOD}_k(A)$, if and only if it satisfies the following conditions;

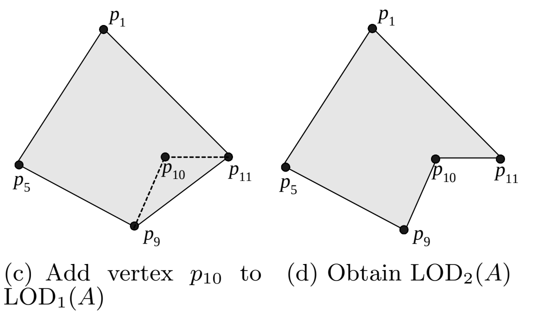
1) $\text{LOD}_k(A) \subset \text{LOD}_{k-1}(A)$ for $k \in \{1,...,N\text{-}1\}$,
2) $\text{LOD}_k(A).V \in \text{LOD}_{k+1}(A).V$ for $k \in \{1,...,N\text{-}1\}$, and
3) $\text{LOD}_0(A)=\text{MBR}(A)$, and $\text{LOD}_N(A)=A$

where $\text{LOD}_{k-1}(A).V$ means the set of vertices of polygon $\text{LOD}_{k-1}(A)$, MBR(A) is the minimum bounding rectangle of A, and N is the number of LODs.

The lowest LOD of polygon A, $\text{LOD}_0(A)$ is defined as the MBR (Minimum Bounding Rectangle) of A with only two vertices, lower and upper corners, while the highest LOD, $\text{LOD}_N(A)$ is the original polygon itself. And $\text{LOD}_1(A)$ is defined as the convex hull of A. The intermediate LODs of A are defined in a recursive way. Starting from $\text{LOD}_1(A)$, the (k+1)-th LOD of A is obtained from $\text{LOD}_k(A)$ by adding a certain number of vertices with a proper policy.



(a) Compute area of $\triangle(p_i,p,p_{i+1})$ (b) Select maximum area



(c) Add vertex p_{10} to $\text{LOD}_1(A)$ (d) Obtain $\text{LOD}_2(A)$

Figure 1: Example for adding a vertex

We derive $\text{LOD}_{k+1}(A)$ from $\text{LOD}_k(A)$ by selecting a vertex among the vertices of the original polygon, which do not belong to the k-th LOD of A ($p_i \in A.V - \text{LOD}_k.V$). p_i is included in $\text{LOD}_{k+1}(A).V$ if 1) the reduced area from $\text{LOD}_k(A)$ to $\text{LOD}_{k+1}(A)$ be maximum and 2) $\angle(p_{i-1},p_i,p_{i+1}) < 180°$. By reducing the area of $\text{LOD}_{k+1}(A)$, we decrease the probability of overlapping between polygons. The second condition is to ensure the first condition of the definition on the progressive polygon representation. In figure 1-a, p_4, p_7, and p_{13} are excluded from the selection for this reason. The procedure is explained in algorithm 1.

Algorithm 1: Algorithm for adding a vertex

```
  input  : $\text{LOD}_1(A)$ and $\text{LOD}_k(A)$ // $A=\{p_1,p_2,...,p_m\}$
  output: $\text{LOD}_{k+1}(A)$
1 begin
2     $v \leftarrow p_i \in \text{LOD}_k(A).V$ such that the area of
      $\triangle(p_{i-1},p_i,p_{i+1})$ is maximum and $\angle(p_{i-1},p_i,p_{i+1}) <$
      $180°$; // $\text{LOD}_k(A).V$ is a set of vertices of $\text{LOD}_k(A)$
3     $\text{LOD}_{k+1}(A) \leftarrow$ Add $v$ to $\text{LOD}_k(A)$;
4 end
```

2.2 Progressive transfer of polygon data

Starting from the lowest LOD (LOD_0), a polygon is transferred to DSMS in a progressive way through a simplification module as shown in figure 2. Multiple LODs are transferred to DSMS, and some of them are discarded by DSMS if they are not overlapping with other polygons in the sliding window. The data format of polygon for the progressive transfer is depicted in table 1.

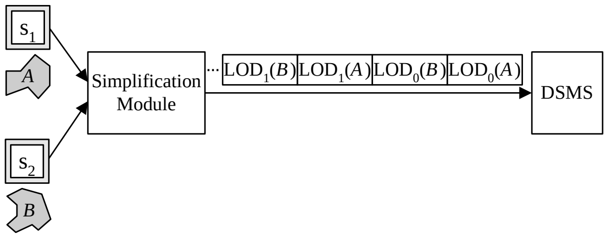


Figure 2: Progressive transfer of polygon to DSMS

Table 1: Message format of polygon for progressive transfer

Element			Size	Description
object id			4 B	Polygon ID
Timestamp			4 B	Valid time of polygon
Levels of Detail			2 B	
♯ of point sets			2 B	Number of consecutive point sets to add
point set	start point		4 B	Point ID of the lower LOD to add a point set
	♯ of points		2 B	Number of additional point to insert
	point	(id,x,y)	12 B	point ID and coordinates

The message format is defined to implement the algorithm for adding a vertex. When a DSMS receives a message of given LOD, it reconstructs a polygon of the specified LOD with the lower LOD and the received message. The message contains the timestamp that the data has been collected, the LOD number, and the geometry to derive the polygon of the specified LOD. The geometry data contains vertex lists to add to $\text{LOD}_k(A)$.

3. PROGRESSIVE SPATIAL JOIN WITH MULTIPLE LODS

The key idea of spatial join processing for polygonal data stream is based on the multiple LODs discussed in the previous section. From this idea, we derive following property, which is very helpful to process spatial join for polygon data stream in an efficient way and improve the performance and accuracy. It describes the conditions that the resulting pairs of spatial join for polygon data stream should satisfy.

Property Conditions of resulting pairs of spatial join on polygon data stream

When two polygons A and B are transferred to the sliding window W of DSMS, they must satisfy the following conditions to be the answer of spatial join;

1) $\mathrm{LOD}_0(A) \cap \mathrm{LOD}_0(B) \neq \varnothing$,
2) If $\mathrm{LOD}_{k-1}(A) \cap \mathrm{LOD}_{k-1}(B) \neq \varnothing$, then $\mathrm{LOD}_k(A) \cap \mathrm{LOD}_k(B) \neq \varnothing$, $k \in \{1,...,N\}$, where N is the number of LOD, and
3) $\mathrm{LOD}_k(A)$ and $\mathrm{LOD}_k(B)$ should be in W.

The first and second conditions are explained by the example in figure 3, which shows a progressive spatial join from the lowest LOD of two polygons. If a pair of two polygons does not intersect at a certain LOD, then we do not need further check of the pair. In the example depicted in figure 3, we stop the evaluation at LOD_2 because two polygons do not intersect any more from LOD_2. This implies that we save a fraction of processing by removing unnecessary pairs of polygon from the candidate pairs.

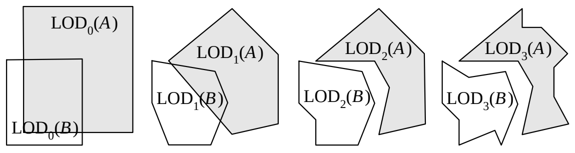


Figure 3: Example of spatial join

The third condition is obvious since the spatial join for polygon data stream is performed only with the objects in the sliding window and the rest objects are unfortunately ignored by the spatial join processing.

We propose the algorithm of spatial join processing for polygon stream. For the reason of convenience, we assume the followings;

- the geometry of polygon is represented by multiple LODs as explained by table 1 in section 2.2,
- the multiple LODs are sent to a single sliding window of DSMS, and the sliding window size is limited.

Algorithm 2 shows the progressive spatial join for polygon stream and the notations are defined as table 2.

4. EXPERIMENT

In this section, we study the performance and accuracy of the method with real [1] and synthetic data sets. Each set contains 2,000 polygons which consist of 30 points.

Table 2: Notations for algorithm 2

Notation	Description
S	Data stream
A,B	Polygons from stream S
W	Sliding window of DSMS
$\mathrm{M}_k(A)$	Message containing the additional geometry for the k-th LOD of A
$\mathrm{M}_k(A).P$	Point list in $\mathrm{M}_k(A)$
$\mathrm{LOD}_k(A)$	k-th LOD of polygon A
n_f	Number of objects removed from W
L_{result}	Candidates set

Algorithm 2: Progressive Spatial Join

```
   input : $M_k(A)$
1  begin
2      $M_k(A)(\in S)$ of polygon $A$ is transferred to DSMS.
3      if $LOD_{k-1}(A) \in W$ then
4          Compute additional size after $M_k(A)$ is added to
           $LOD_{k-1}(A)$;
5          if $W$ is overflowed by extended size then
6              Flush $n_f$ numbers of objects of low priority
               from $W$;
7          end
8          $LOD_k(A) = LOD_{k-1}(A) \cup M_k(A).P$;
9          Find $LOD_i(B)$ which intersects with
           $LOD_{k-1}(A)$ in $W$ from $L_{result}$;
10         if $LOD_i(B)$ exists and $LOD_k(A) \cap$
           $LOD_i(B) = \varnothing$ then
11             Remove $(A,B)$ from $L_{result}$;
12         end
13     end
14     else
15         if $k=0$ then
16             Insert $LOD_0(A)$ into $W$ if $W$ has enough
               space. If $W$ is overflowed, then same way in
               line 5-7;
17         end
18         else
19             Delete $M_k(A)$;
20         end
21     end
22 end
```

We set up several parameters for our experiments. The number of LODs for simplification N, varies from one to five. When N=1, it corresponds with the conventional spatial join method which delivers the entire geometry of polygon to DSMS at the same time. When N=2, the first and last LODs represent the MBR of polygon and the original polygon, respectively. The arrival times are set up by the exponential distribution. The size of sliding window, n_W, varies between 30 and 600 kilobytes (Kb). When the sliding window is overflowed, we select the polygon object that has the least intersection with other polygon in the sliding window to flush it from the sliding window.

In this paper, we discuss on the results of experiments and analysis in terms of CPU time for geometric computation costs and the recall rate of the join which is defined as the fraction of the number of pairs selected by progressive spatial join method over the number of all pairs satisfying the spatial join. Note that we carried out the experiments with Intel 2.33GHZ Processor of 3.25GB RAM.

4.1 Geometric processing cost

Figure 4 shows the average CPU time of spatial join as window size n_W changes. The result of experiments shows that the processing cost with multiple LODs becomes almost constant as the window size is larger than 90 Kb regardless of data sets, while the conventional method with non-progressive approach shows a significant increase of processing cost as the window size increases. It means that we reduce a considerable amount of CPU cost by removing unnecessary polygons from sliding window and simplifying the polygons. We can observe similar result from synthetic data stream, so we exclude the result from the paper.

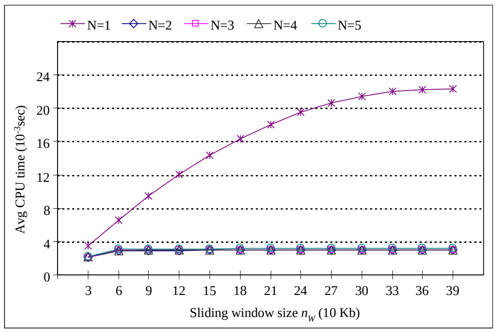
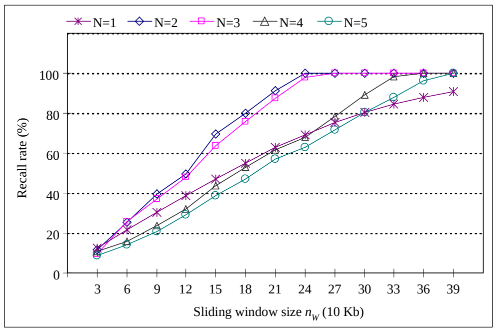
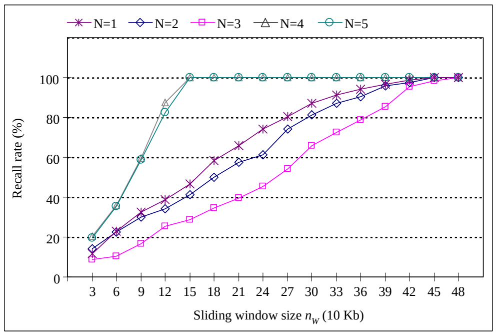


Figure 4: Avg. CPU time as n_W changes(weather)



(a) Weather data stream



(b) Synthetic data stream

Figure 5: Recall rate on sliding window n_W

4.2 Recall rate

Figure 5 shows the recall rates of join results with different numbers of LODs, N. The results show that the recall rates are nearly optimal when $N = 2$ or $N = 3$ with the weather data stream (figure 5-a). However, the recall rate for the synthetic data is improved with larger numbers of LODs, such as $N = 4$ or $N = 5$ as shown in figure 5-b.

The reason why two input data streams show different results of recall rates is closely related with the geometrical properties of polygon of data stream. Most simplified polygons of intermediate LODs in the weather data stream have little dead spaces. For this reason, most polygons that do not intersect with other polygons are filtered even at a low LODs such as MBRs and convex hulls from sliding window.

However, most polygons of intermediate LODs in synthetic data stream have relatively large dead space. In this case, we cannot determine whether two original polygons intersects or not with their MBRs or convex hulls. We need a large number of LOD to improve the recall rate when the original polygons have large dead spaces.

5. CONCLUSION AND FUTURE WORK

Handling spatial data stream is one of most crucial functions in DSMS. When we process spatial join for polygon data stream, there are two critical problems. First, the large size of polygon may significantly degrade the accuracy of spatial join on data stream due to the limited size of sliding window. Second, the cost of spatial join processing is quite expensive due to the geometry computation of polygons. In this paper, we proposed a spatial join processing method for polygon streams. Our method is based on progressive representation and transfer of polygon to DSMS to overcome two critical problems. And we carried out several experiments to analyze the performance and accuracy of our method. As discussed in section 4, the accuracy of spatial join processing for polygonal data stream is closely related with geometric complexity of polygons. We need further work to analyze the relationship between the complexity of polygon and recall rates.

Acknowledgement

This work was supported in part by the Brain Korea 21 Project in 2011 and a grant (06KLSGB01) from Cutting-edge Urban Development - Korean Land Spatialization Research Project funded by Ministry of Land, Transport and Maritime Affairs.

6. REFERENCES

[1] United States Department of Commerce. National oceanic and atmospheric administration(noaa). http://www.noaa.gov/.

[2] W. H. Tok, S. Bressan, and M.-L. Lee. Progressive spatial join. In *Proceedings of the 18th International Conference on Scientific and Statistical Database Management*, pages 353–358. IEEE, July 2006.

[3] X. Zhu, H. Gupta, and B. Tang. Join of multiple data streams in sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 21(12):1722–1736, December 2009.