Journal of Engineering and Computer Innovations Vol. 3(1), pp. 1-10, February 2012
Available online at http://www.academicjournals.org/JECI
DOI: 10.5897/JECI11.008
ISSN 2141-6508 ©2012 Academic Journals

Full Length Research Paper

Optimization of spatial join using constraints based-clustering techniques

V. Pattabiraman

School of Computing Science and Engineering, VIT University - Chennai - 600 048, Tamil Nadu, India.
E-mail: pattabiraman.v@vit.ac.in.

Accepted 19 December, 2011

Spatial joins are used to combine the spatial objects. The efficient processing depends upon the spatial queries. The execution time and input/output (I/O) time of spatial queries are crucial, because the spatial objects are very large and have several relations. In this article, we use several techniques to improve the efficiency of the spatial join; 1. We use R*-trees for spatial queries since R*-trees are very suitable for supporting spatial queries as it is one of the efficient member of R-tree family; 2. The different shapes namely point, line, polygon and rectangle are used for isolating and clustering the spatial objects; 3. We use scales with the shapes for spatial distribution. We also present several techniques for improving its execution time with respect to the central processing unit (CPU) and I/O-time. In the proposed constraints based spatial join algorithm, total execution time is improved compared with the existing approach in order of magnitude. Using a buffer of reasonable size, the I/O time is optimal. The performance of the various approaches is investigated with the synthesized and real data set and the experimental results are compared with the large data sets from real applications.

Key words: Spatial data mining, spatial clustering, spatial queries, spatial join.

INTRODUCTION

Spatial join operation is used to cluster two or more dataset with respect to a spatial predicate. Predicate can be a combination of direction, distance, and topological relations of spatial objects. In non spatial join, the joining attributes must be of the same type and in spatial join they can be of different types. Each spatial attribute is represented by its minimum bounding rectangles (MBR).

A typical example of spatial join is “Find all pair of rivers and cities that intersect”. For example in Figure 1, the result of join between the set of rivers {R1, R2} and cities {C1, C2, C3, C4, C5} is { (R1, C1), (R2, C5)} (Shashi and Sanjay, 2003).

The spatial objects can be movable and immovable objects. Hash based algorithms focus only on natural join and equi-join. Since spatial objects are multidimensional data, we need new efficient spatial join algorithm. The methods for computing the spatial join are discussed in great detail for quad trees and similar access methods (Shou et al., 2003).

Key concepts

The two steps involved in spatial join are filter step and refine step. In filtering step, tuples whose minimum bounding rectangle (MBR) overlaps with query region are determined. This step is not computationally expensive but it requires at most four computations to determine rectangles intersection. In refine step, the tuples which passed the filter step is fed to the refinement step where exact spatial representation is used and spatial predicate is checked on these spatial representations. Refinement step is computationally expensive, but the number of tuples it processed in this step is less, due to initial filter step. Spatial join algorithm can be classified into three categories: nested Loop, tree matching and partition-based spatial merge join (Brinkhoff et al., 1993).

In general the spatial objects are of two types, namely, movable objects and immovable objects. The process that includes movable objects are like to identify the cell

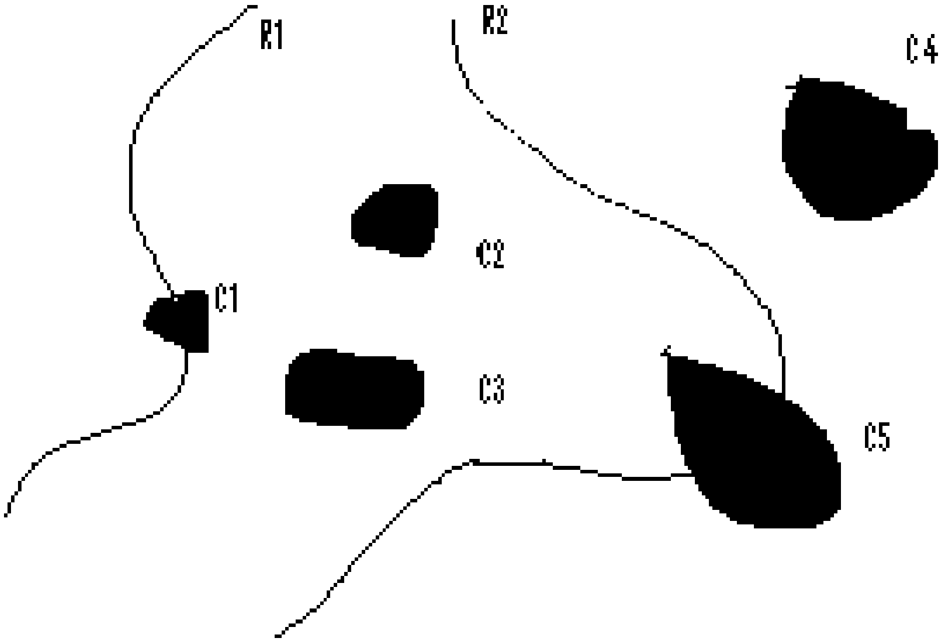

Figure 1. Example of spatial joins (Shashi and Sanjay, 2003).

phone towers of the respective cellular network service provider and in process of immovable object are to find identify the ideal object in the multi spectral image (spatial data). For the movable objects the spatial queries are classified into single scan queries and multi-scan queries.

In this paper, we propose the spatial joins for immovable objects like to find the banks that are nearer to the PSG College of Arts and Science. Initially, with the help of the constraints based spatial clustering algorithm the banks are identified within Coimbatore. Here we use R*-trees for spatial database querying.

Clustering introduction

Clustering is the classification of objects into different groups, or more precisely, the partitioning of a data set into subset (clusters), so that the data in each subset (ideally) shares some common trait-often proximity according to some defined distance measure. Data clustering is a common technique for statistical data analysis which is used in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics. The computational task of classifying the data set into k clusters is often referred to as k-clustering.

Types of clustering

Data clustering algorithms can be:

i. Hierarchical algorithms: Find successive clusters using previously established clusters. Hierarchical algorithms can be agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.

ii. Partitioned algorithms: Typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering.

iii. Density-based clustering algorithms: Are devised to discover arbitrary-shaped clusters. In this approach, a cluster is regarded as a region in which the density of data objects exceeds a threshold. DBSCAN and OPTICS are two typical algorithms of this kind.

iv. Two-way clustering, co-clustering or biclustering: Are clustering methods where not only the objects are clustered but also the features of the objects, that is, if the data is represented in a data matrix, the rows and columns are clustered simultaneously.

Another important distinction is whether the clustering uses symmetric or asymmetric distances. A property of Euclidean space is that distances are symmetric (the distance from object A to B is the same as the distance from B to A).

Spatial database system is a database system for managing spatial data. There is a rapid growth in number and the size of spatial databases for applications such as geo-marketing, traffic control, and environmental studies (Orenstein, 1986). Spatial data mining, or knowledge discovery in spatial databases refers to the extraction from spatial databases of implicit knowledge, spatial relations, or other patterns that are not explicitly stored (Shashi and Sanjay, 2003).

Clustering analysis for data in a 2-D space is considered as spatial data mining and its applications are geographic information systems, pattern recognition, medical imaging, marketing analysis, weather forecasting, etc. Clustering in spatial data is an active research area and most of the research focus on effectiveness and scalability.

RELATED WORK

Historical background

In 1986, Orenstein used grid based technique to perform spatial join. It is the first known technique to solve spatial join operation. Using a multidimensional grid, spaces are divided into smaller blocks, known as pixels. Then, z-ordering is used to order the pixels. Each object is approximated by the pixels which intersect with its MBR. As pixels are ordered by z-ordering, each object is represented by a set of z-values which are one-dimensional. Now, any one-dimensional indexing (e.g., B+-tree) can be used to sort them and by using sort-merge, spatial join operation is done. The performance of this technique solely depends on the granularity of the grids. The finer grids gives the accurate results but with higher memory consumption. Later on to remedy this problem, multidimensional indices (e.g., R-tree), which can directly handle spatial data, were devised. Various new spatial join algorithms (e.g., R-tree join, sort and match, spatial hash join, slot index hash join etc.) based on multi-dimensional index appeared (Brinkhoff et al., 1993).

Base work

In the base paper, data file is in (xls) format and used for the spatial join1 algorithm. Data is read into buffers with different last recently used (LRU) size in bytes 0, 8, 32, 128, 256 and 512 and compared with data in the size 1KB, 2KB, 4KB and 8KB (while inserting in the buffer) to find the optimal comparisions time.

In spatial join2 algorithm, before taking the buffer for the comparisions, apply the sorting technique to arrange the data(line shape) in order. For example each areas are identified with the unique identified code namely FID and it is sorted in order using nearby locations with detailed information like name of the place. Then use the spatial join1 algorithm for further process.

In spatial join3 algorithm, based on spatial ordering this algorithm creates a sequence of pairs of intersectional rectangles. Obviously, this sequence can also be used to determine the read schedule of the spatial join.

Partition clustering

K-means clustering

The K-means algorithm assigns each point to the cluster whose center (also called centroid) is the nearest. The center is the average of all the points in the cluster, that is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster. For example, a data set has three dimensions and the cluster has two points: $X = (x_1, x_2, x_3)$ and $Y = (y_1, y_2, y_3)$. Then, the centroid Z becomes $Z = (z_1, z_2, z_3)$, where $z_1 = (x_1 + y_1)/2$ and $z_2 = (x_2 + y_2)/2$ and $z_3 = (x_3 + y_3)/2$.

The algorithm steps are:

i. Choose the number of clusters, k.
ii. Randomly generate k clusters and determine the cluster centers, or directly generate k random points as cluster centers.
iii. Assign each point to the nearest cluster center.
iv. Recompute the new cluster centers.
v. Repeat the two previous steps until some convergence criterion is met (usually that the assignment has not changed).

The main advantages of this algorithm are its simplicity and speed which allows it to run on large datasets. Its disadvantages are not yielding the same result with each execution, since the resulting clusters depend on the initial random assignments. It minimizes intra-cluster variance, but does not ensure that the result with global minimum of variance.

Quality threshold (QT) clustering algorithm

QT clustering is an alternative method of partitioning data, invented for gene clustering. It requires more computing power than k-means, but need not to specify the number of clusters in prior, and also it returns the same result when several executions takes place. The algorithm is:

i. The user chooses a maximum diameter for clusters.
ii. Build a candidate cluster for each point by including the closest point, the next closest, and so on, until the diameter of the cluster surpasses the threshold.
iii. Save the candidate cluster with the most points as the first true cluster, and remove all points in the cluster from further consideration. Must clarify what happens if more than 1 cluster has the maximum number of points?
iv. Recurse with the reduced set of points.

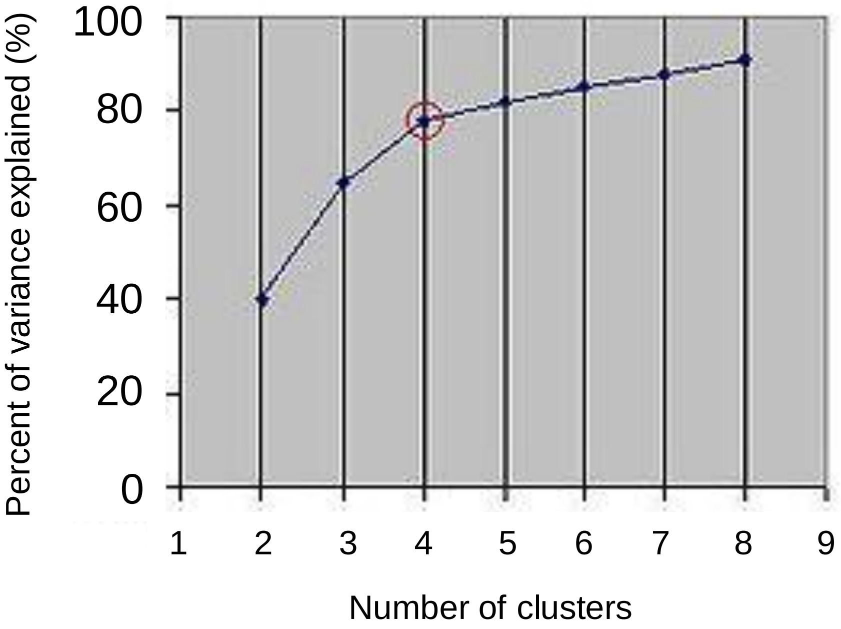

Figure 2. Percent of variance with the number of clusters.

The distance between a point and a group of points is computed using complete linkage, that is, as the maximum distance from the point to any member of the group.

Locality-sensitive hashing

Locality-sensitive hashing can be used for clustering. Feature space vectors are sets, and the metric used is the Jaccard distance. The feature space can be considered high-dimensional. The min-wise independent permutations LSH scheme (sometimes MinHash) is then used to put similar items into buckets. With just one set of hashing methods, there are only clusters of very similar elements. By seeding the hash functions several times (e.g. 20), it is possible to get bigger clusters.

Graph-theoretic methods

Formal concept analysis is a technique for generating clusters of objects and attributes, giving a bipartite graph representation for relations between the objects and attributes. Other methods for generating overlapping clusters are also discussed.

Determining the number of clusters

If the number of the clusters is not apparent from prior knowledge, it should be chosen in some way (Figure 2). Several methods for this have been suggested within the statistical literature where one rule of thumb sets the number to:

$$k \approx (n/2)^{1/2} \qquad (1)$$

Where n is the number of objects (data points).

The "elbow" is indicated by the red circle. The number of clusters chosen should be 4. Another thumb rule looks at the percentage of variance explained as a function of the number of clusters: We should choose a number of clusters so that adding another cluster does not give much better modeling of the data. More precisely, if we graph the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters chosen at this point is "elbow criterion". This “elbow” cannot always be unambiguously identified. Percentage of variance explained is the ratio of the between-group variance to the total variance. A slight variation of this method plots the curvature within group variance.

Other ways to determine the number of clusters use Akaike’s information criterion (AIC) or Bayesian information criterion (BIC), if it is possible to make a likelihood function for the clustering model. For example, the k-means model is almost a Gaussian mixture model and thus also determine AIC and BIC values.

In a Geographic Information System (GIS) application, studying the movement of pedestrians to identify optimal bank machine placements, for example, the

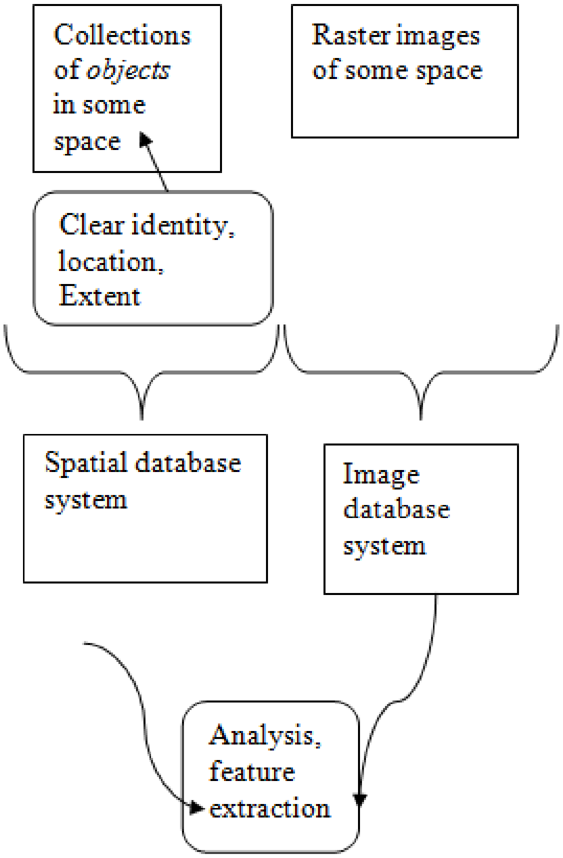
Collections of *objects* in some space
Raster images of some space
Clear identity, location, Extent
Spatial database system
Image database system
Analysis, feature extraction

Figure 3. An example of a spatial database.

Table 1. Sample spatial data set.

FID	Shape	Division	School no	School name
0	Point	1	1	Alvernia Matric School
1	Point	6	2	Alwyn English Primary School
2	Point	12	3	Avila Matric School
3	Point	10	4	A.L.G Matriculation School
4	Point	6	5	Bharathi Vidya Bhavan

A spatial database may contain.

presence of a highway hinders the movement of pedestrians and should be considered as an obstacle. To the best of our knowledge, the following clustering algorithms for clustering spatial data with the constraints are DBRS, DBRS+, COD_CLARNS, AUTOCLUST, AUTOCLUST+, DBCluc, IKSCOC, GKSCOC and PSOCOC.

SPATIAL DATABASE

Spatial database is a system related to some space (Shashi and Sanjay, 2003). It should contain the collection of objects related to space of different types (Figure 3).

The different objects of spatial database

1. Point: ● represents a single location like city or particular place (Table 1)
2. Line: 〰 moving through space, connections in space (River, Highways).
3. Region: is an abstraction of ⬬ ⬟ an object with extent (forest, Lake, City).

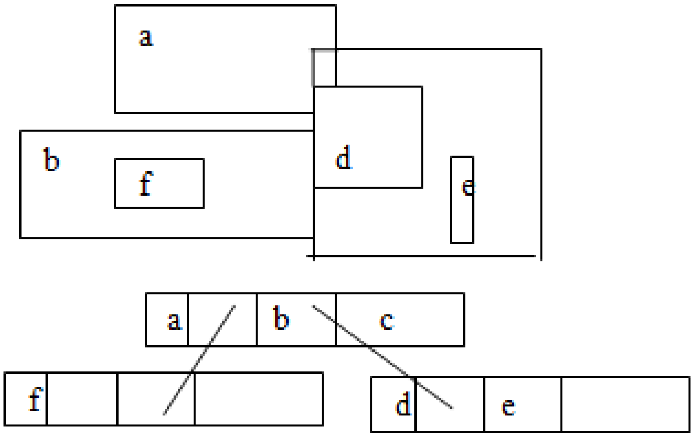

Figure 4. Data structure of R-tree.

Fundamental operations on spatial data (Algebra): i. Spatial selection, ii. Spatial join, iii. Spatial function application, iv. Other set operations,

Spatial selection - selection based on a spatial predicate. Example queries are given below:

a. “Find all schools in Coimbatore.” – School select [center inside Coimbatore],
b. “Find all rivers intersecting a query window.” - rivers select [route intersects Window],
c. “Find all big Municipal cities no more than 100 km from Coimbatore.” - cities select[dist(center, Coimbatore 100 and pop > 500 000].

Spatial join - join based on a predicate comparing spatial attribute values (Mamoulis, 2001). Example queries are given below:

a. “Combine Municipal cities with their districts.” – municipal cities districts join[center inside area],
b. “For each river, find all municipals cities within less than 50 km.” – municipal cities rivers join[dist(center, route) < 50] a REGION value and a POINT value.

Spatial function application - We use operations of a spatial algebra computing new SDT values by regions lines and lines intersection:

a. In selection conditions.
b. Object algebra operators allow one to apply functions to each member of a set: i. Filter operator (FAD), ii. Replace, iii. Map and iv. Extend.

Example queries are given below:

“For each river going through Coimbatore, return the name, the part inside Coimbatore and the length of that part.” - rivers select[route intersects Coimbatore] extend[intersection(route, Coimbatore) {part}] extend[length(part) {plength}] project[rname, part, length] (Ralf, 1994).

Spatial query processing

Since spatial database contains different objects, we need excellent access methods to handle the data. In traditional cases minimum bounding rectangle (MBR) concept was used (An and Sivasubaramainam 2001). In this case, two techniques are used:

(i) Filter step: Find all objects whose MBR intersects the query rectangle.
(ii) Refinement step: For those objects, check whether they really fulfill the query condition (if necessary, make use of the exact representation).

The basic idea of these methods is to decompose the data space into non-overlapping cells which can be computed by recursively cutting the space into two or four parts of equal size (Corral, 2000). Cells can be identified by a location code, called z-value, and a size code, called level. In R*-trees also, we should not give the redundant values (Guttmann, 1984).

R*-Tree

An R-tree is a B*-tree like access method that stores multidimensional rectangles as complete objects without clipping them or transforming them to higher dimensional points. Until now, the most efficient variant of the R-trees is experimentally shown to be the R*-tree. The R*-tree uses more sophisticated insertion and splitting algorithms than the original R-tree (Figure 4). However, there is

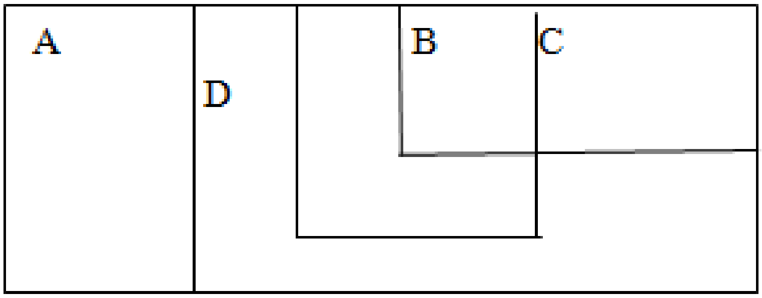

Figure 5. Example of spatial joins.

almost no difference in the data structure (Samet, 1990).

Advantage: Spatial object (or key) in a single bucket.
Disadvantage: Multiple search paths due to overlapping regions.

Spatial joins using R*-tree

The Central Processing Unit (CPU) time is calculated by using the floating point comparison.
Algorithm: (An and Sivasubaramainam, 2001)

```
SpatialJoinl (R, S: R_Node); (* height of R is equal height of S *)
FOR (all Es € S) DO
FOR (all $E_R$ € R with $E_R$.rect o Es.rect # Ǿ) DO
IF (R is a leaf page) THEN
    (* (S is also a leaf page)*)
    Output $E_R$,$E_S$
ELSE
    ReadPage($E_R$.ref); ReadPage(Es.ref);
    SpatialJoinl $E_R$.ref, $E_S$.ref
END
END
END
END SpatialJoin 1;
```

Here B and C are joined with D.

We have to consider each and every point, that is, A with B, C, D and B with A, B, and C and so on (Figure 5). This will take the comparison more (n^2). So this is not effective for more spatial joins. Sorting the objects belonging to their category can be done in prior to Spatial Joins for time optimization of CPU process.

The proposed spatial join algorithm:

Step 1: Read the spatial objects with the constraints.
Step 2: Cluster the spatial objects based on the constraints.
Step 3: Sort the clustered spatial objects.
Step 4: Compare the nodes of the spatial objects with the relationship.

The objects may be in single relation or multiple relations (Manli, 2005). For example, combining the highways in Coimbatore city. This is simple relation. Combining the highways that are connecting the universities is multiple relations. Likewise the complexity may get increased.

CONSTRAINTS BASED SPATIAL JOIN

The proposed spatial joins with the constraints manipulate whole sets of spatial objects in a special way. The spatial join operation is a conceptual unit which aims to address the constraints based spatial objects.

K-medoids clustering under spatial data with obstacle constraints

This is because in spatial data, there are two kinds of objects namely movable and immovable objects. To identify the moving and non-moving objects, here we use K-medoids clustering with the obstacle using edge detection method. This helps to cluster the data with the obstacle constraints to identify movable and immovable objects under spatial data (images). It is easy and less time consumption (An and Sivasubaramainam, 2001). K-Medoids clustering generates the clusters with much related data.

Spatial joins for datasets

After K-Medoids, it is easy to join our spatial data. By using the modified spatial join algorithm we can combine our datasets (Faloutsos, 2000).

Spatial joins for images

If the images are spatial images then it is possible to cluster by using image clustering algorithms. After that,

Table 2. CPU and I/O time tunning.

The data were implemented by taking the buffer size of 1024 and the objects are all point shape					
		Size of pages			
		1 KByte	2 KByte	4 KByte	8 Kbyte
Size of buffer	0	24,727	12,479	5720	2837
	8	20318	12010	5720	2837
	32	13803	9589	5454	2822
	128	11359	6299	4474	2676
	512	10372	4964	2768	2181
	1024	9589	4372	1084	1027

we can use the modified Spatial join algorithm. Hence, to identify the movable objects we can use K-NN (K-nearest neighnour algorithm) and for immovable objects Spatial Joins Algorithm. The above process are most essential for applying spatial joins under the spatial data (images).

Spatial joins with points

With the help of clustering algorithm in data mining, it is possible to group the data under spatial objects (images). Consequently, we can use K-mediods constraint based clustering algorithms under the spatial data to find the cluster with the shapes like (point, line, polygon) based on the constraints like river, mountain, highway and buildings (Koudas, 2001). After finding the clusters under the spatial data we can apply spatial joins for easy identification of locations in geographical area (Yannis, 2005). The spatial joins are applicable only for the idle objects namely city, school, college and banks, etc. in the spatial objects. In base paper, they did not focused on clustering by assuming that the cluster already exist. Hence, straight away they have taken towards spatial joins. But in our work first we use clustering technique to identify the clusters under the spatial data, then it is taken for the spatial query processing and also for spatial joins.

The only data which is taken for the spatial join in the existing papers is the line objects. But we have taken various objects points namely: city, school, college and banks etc., in the spatial objects (image) to find the spatial joins between these object points. Likewise in future, it is possible to take the following combinations of spatial objects like point-point, line-line, polygon-polygon, point-line, line-point, point-polygon, polygon-line to find the spatial joins between them (Mamoulis, 2001).

In the base paper, without using algorithms with the properties, they have taken two files namely R and S for R* tree comparison. But it has high time consumption for comparison. The proposed spatial join algorithms will take the optimum time for the comparison. If we are applying the above strategies under the spatial data (image), it is very easy to identify or locate the exact geographical area which is needed for the user.

RESULTS AND DISCUSSION

Performance comparison of CPU and I/O time tunning

Table 2 shows the CPU and I/O time tuning based on the buffer size 1024KB and the objects are all point shape. In addition to preserve spatial locality in the buffer, this approach can be used without any extra cost. Hence, call this approach as local plane-sweep order and the corresponding join algorithm as spatial join3. Note that it is assumed in the algorithm sorted intersection test does not use sequences of rectangles as introduced originally, but sequences of entries for input and output and also assume that the spatial data to be taken for spatial join after applying the constraints based spatial clustering. Compared with the existing algorithm, the proposed method provide less complexity with the best improvement of CPU time using R*-tree spatial joins with the constraints.

Figure 6 shows the performance comparison chart based on the size of the pages with the size of the buffer. Comparing with the existing method, the proposed one gives better result without any extra cost and also the chart represents the multiples of bytes in terms of size of the pages with the buffer size.

One of the key applications of spatial join is to find all the objects which either intersect or overlap with each other. Some variants of spatial join (e.g., distance join) are used in data mining for data analysis and clustering. It can also be used to process closest-pairs query, k-nearest neighbors query, and ϵ-distance query. Figure 7 shows the comparison chart of processor based on the size of the pages with the size of the buffer. The Anthon 64FX, Pentium 4 (III, “Prescott”) shows competent performance compare to other processor.

FUTURE WORK

There are some issues in spatial join that require further attention from research community. For processing spatial join queries, we usually follow filter and refine step

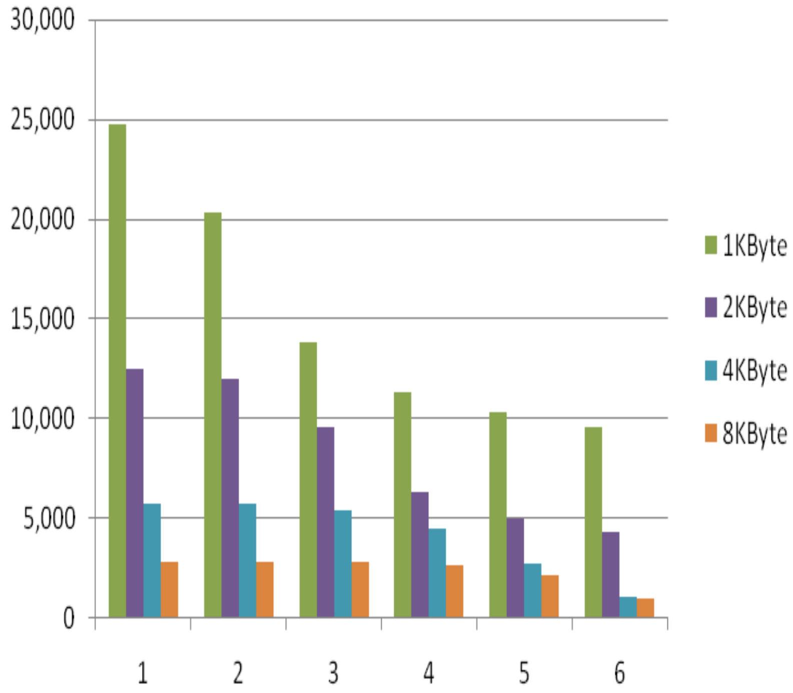

Figure 6. Size of pages in buffer vs. buffer size.

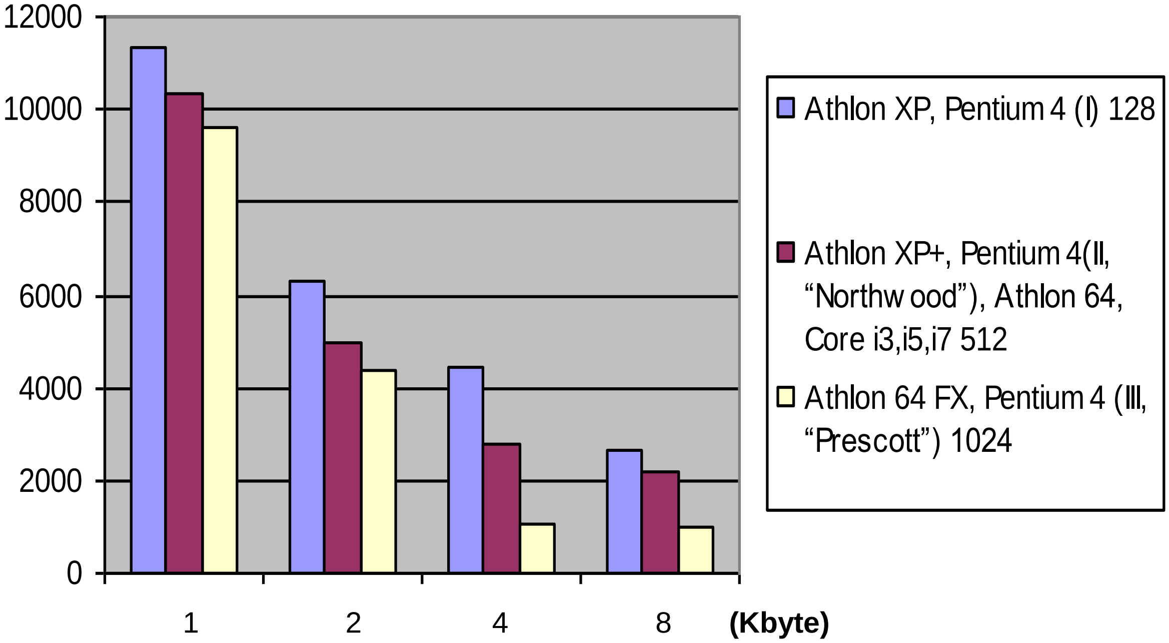

Figure 7. Comparison of processor, size of buffer and pages.

in order. In some cases, some variants of this (e.g., interleaving) may give us more benefit. We can explore where probable variants can be beneficial and what information we need to collect for this. Although, intersection join algorithms (e.g., R-tree join) can be directly extended for other types (e.g., distance join), it causes inefficient performance benefit. Various optimization techniques can be applied to rectify this.

Extending existing intersection join algorithms with various optimization criteria to other domain, it will be an interesting area for research.

Conclusion

In this paper, the discussion was focused on the spatial join effects with the constraints-based spatial data without any extra cost. The constraint-based spatial clustering reduces the time taken to identify the required objects under the spatial image. The R*-tree concept reduce the number of search pages to combine spatial objects. By using this, CPU utilization time increases, the number of comparison of spatial objects can be reduced and also reduces the I/O time.

REFERENCES

An NY, Sivasubaramainam A (2001). Selectivity estimation for Spatial Joins” Proceedings of the IEEE ICDE Conference. pp. 368-375.

Brinkhoff T. Kriege H, Seeger B (1993). Efficient processing of spatial joins using R-trees” Proceeding of ACM SIGMOD Conference. pp. 237-246.

Brinkhoff T, Kriegel H, Seeger B (1996). Parallel processing of spatial joins using R-trees” Proceeding of ICDE Conference. pp. 258-265.

Corral A, Manolopoulos Y, Theodorisdis Y. Vassilakopoulos M (2000). Closest pair queries in spatial databases”, Proceedings of the ACM SIGMOD Conference. pp. 189-200.

Faloutsos C, Seeger B, Traina A, Traina C (2000). Spatial Join Selectivity Using Power Laws”, Proceedings of the ACM SIGMOD Conference, pp. 177-188.

Guttmann A (1984). R-trees: A dynamic index structure for spatial searching”, Proceedings of the ACM SIGMOD Conference. pp. 47-57.

Koudas N, Sevcik K (2000). High Dimensional Similarity Join”, Proceedings of the ACM SIGMOD Conference. pp. 324-335.

Mamoulis N, Papadias D (2001). Multi-way Spatial Joins”, “ACM Transactions on Database Systems. 26(4): 424 - 475.

Mamoulis N, Papadias D (2003). Slot Index Spatial Join”, IEEE Transactions on Knowledge and Data Engineering. 15(1): 211-231.

Manli Z, Dimitris P, Jun Z, Dik Lun L (2005). Top-k Spatial Joins”, IEEE Transactions on Knowledge and Data Engineering, 17(4): 567-579.

Orenstein JA (1986). Spatial Query Processing in an Object-Oriented Database System”, Proceeding of ACM SIGMOD Int. Conf. on Management of Data, Washington D.C. pp. 326-333.

Patel JM, Dewitt DJ (1996). Partition Based Spatial-Merge Join”, Proceedings of ACM SIGMOD Conference. pp. 259-270.

Agrawal R and Srikant R (1994). Fast Algorithms for Mining Association Rules,” Proceeding of Very Large Data Bases Conference. pp. 487-499.

Samet H (1990). The Design and Analysis of Spatial Data Structures”, Addison Wesley, 1990.

Shashi S, Sanjay C (2003). Spatial Databases A Tour”, First Edition, Prentice Hall, 2003.

Shou Y, Mamoulis N, Cao H, Papadis D, Cheung DW (2003). Evaluation of Iceberg Distance Joins”, Proceedings of the Eighth International Symposium on Spatial and Temporal Databases, 2003, pp. 270 - 288.

Yannis RM, Apostolos P, Michel Gr. Vassilakopulous (2005). Spatial Databases, Technologies, Techniques and Trends”, IDEA Group Publishing.

Ralf HG (1994). An Introduction to Spatial Database Systems”, Special Issue on Spatial Database Systems of the VLDB Journal, 3(4): 357-399.