SS-ZG548: ADVANCED DATA MINING

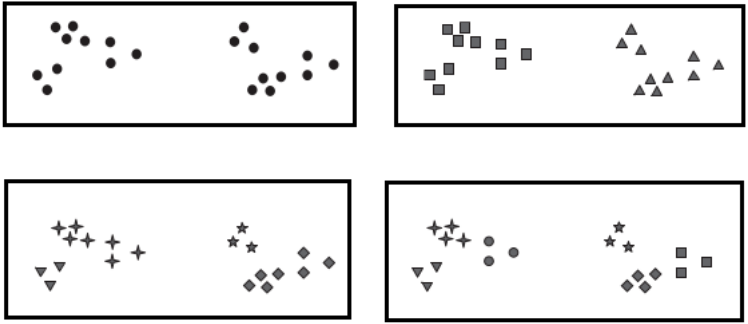
Lecture-05: Incremental Clustering

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems Engineering,
BITS Pilani, Rajasthan-333031 INDIA

Jan 21, 2018 (WILP @ BITS-Pilani Jan-Apr 2018)

Clustering

Grouping data based on their homogeneity (similarity or closeness).



Objects within a group are similar (or related) and are different from the objects in other groups. When it is better?

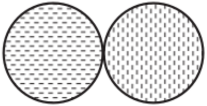
Clustering

- **Unsupervised** in nature (i.e. right answers are not known)
- Clustering is useful to 1) Summarization, 2) Compression, and 3) Efficiently Finding Nearest Neighbors
- **Type:**
  - Hierarchical (nested) versus Partitional
  - Exclusive versus Overlapping versus Fuzzy
  - Complete versus Partial
- **K-means:** This is a prototype-based[1], partitional clustering technique that attempts to find a user-specified number of clusters (K), which are represented by their centroids.

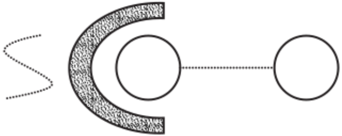
[1] object is closer (more similar) to a prototype

Clustering

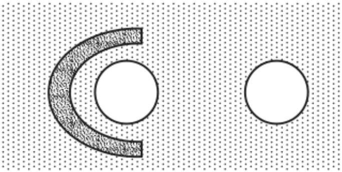
Well-separated clusters.



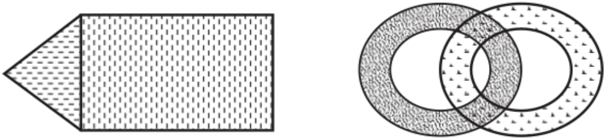
Center-based clusters.



Contiguity-based clusters.



Density-based clusters.



Conceptual clusters.

K-means Algorithm

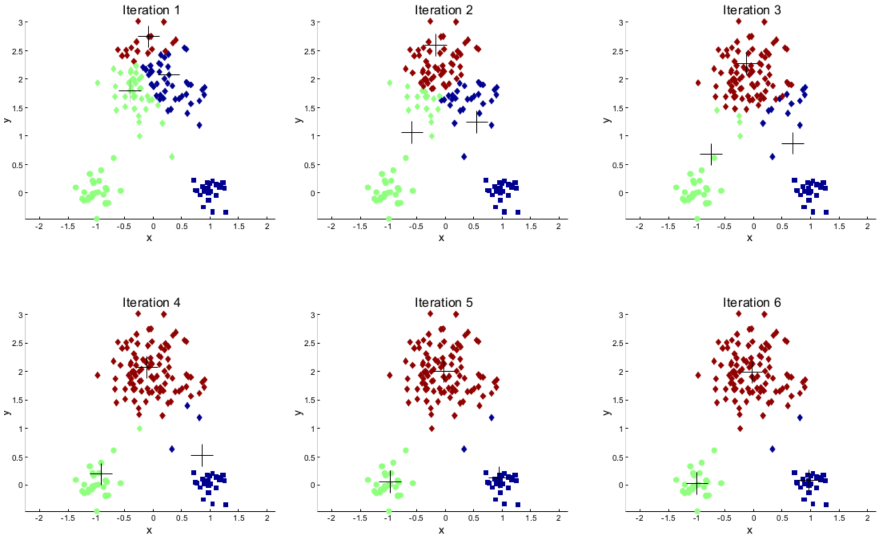
Number of clusters *i.e.* the value of K is provided by the user

Algorithm 0.1: K-means

```
1  Randomly select K points as centroids
2  repeat
3      foreach datum point d_i do
4          Assign d_i to one of the closest centroids
           (thereby forming K clusters)
5      Recompute centroid (mean) for each cluster
6  until The centroids converge;
```

Closeness is measured by **Euclidean distance**, cosine similarity, correlation, Bregman divergence *etc*

K-means in Action



Evaluation of K-means

For a given data set $\{x_a, x_2, ..., x_n\}$, let K-means partitions it in $\{S_1, S_2, ..., s_K\}$ then the objective is

$$\underset{S}{\text{argmin}} \sum_{i=1}^{K} \sum_{x \in S_i} dist^2(x, \mu_i)$$

- Where μ_i corresponds to i^{th} centroid. $\mu_i = \frac{1}{m_i} \sum_{x \in S_i} x$
- Typical choice for *dist* function is Euclidean Distance

How to proceed?

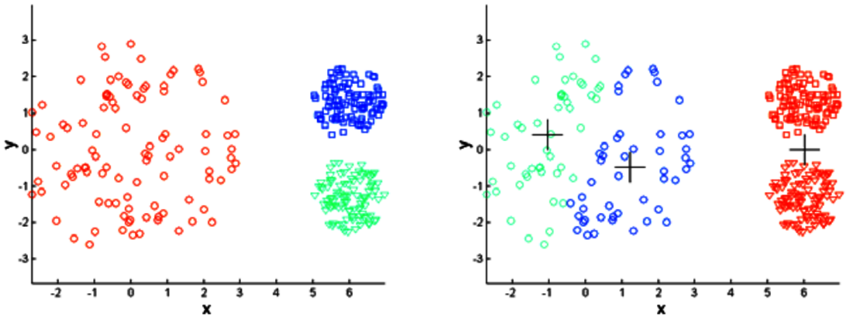
- Choose a K (How? [2])
  - Run K-means algorithm multiple times
  - Choose clusters corresponding to the one that minimized sum of squared error (SSE)
- If $K == n$, no error.
- Good clustering has smaller K

[2] Hamerly, Greg and Elkan, Charles, "Learning the k in k-means", pp 281–288, NIPS-2003

Evaluation of K-means

- **Choosing K:** 1) Domain Knowledge, 2) Preprocessing with another algorithm, 3) Iteration on K
- **Initialization of Centers:** 1) Random point in space, 2) Random point of data, 3) look for dense region, 4) Space uniformly in feature space
- **Cluster Quality:** 1) Diameter of cluster verses Inter-cluster distance, 2) Distance between members of a cluster and the cluster center, 3) Diameter of smallest sphere, 4) Ability to discover hidden patterns

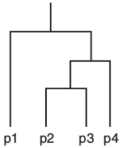
Limitations of K-means

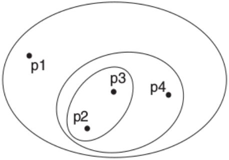


- Has problem when data has
  - Different size clusters
  - Different densities
  - Non-globular shape
- Handling Empty Clusters
- When there are outliers
- Updating Centroids Incrementally

Other Approaches

- **K-Medoids:** chooses data-points as centers and minimizes a sum of pairwise dissimilarities. Resistance to noise and/or outliers
- **Agglomerative Hierarchical Clustering:** repeatedly merging the two closest clusters until a single (Single Link)





- **DBSCAN:** density-based clustering algorithm that produces a partitional clustering, in which the number of clusters is automatically determined by the algorithm.

Important Note:

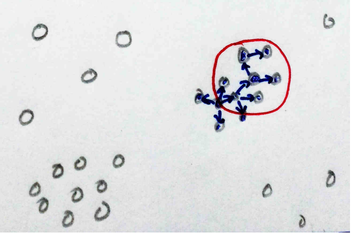
- K-Means and K-NN are different (K nearest neighbors)

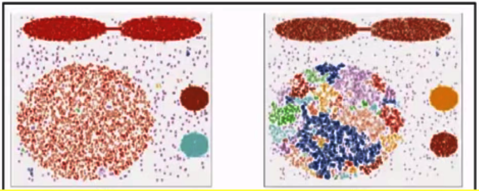
K-NN is a **supervised** approach for **classification**

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a spatial clustering algorithm of KDD96

- Parameters (Eps/MinPts) and points (core/border/noise)
- Uses DFS

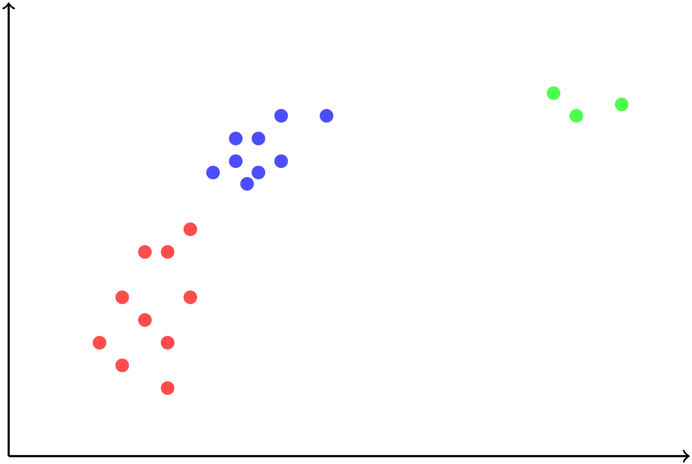




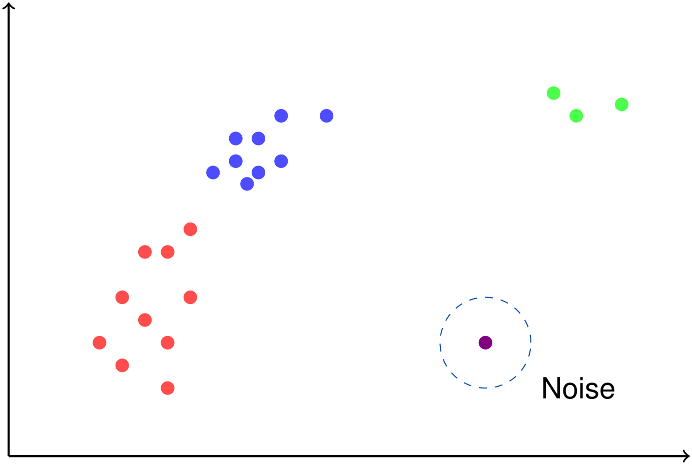
Figures from G. Karypis, E.-H. Han, and V. Kumar, *COMPUTER*, 32(8), 1999

- Advantage: clusters of arbitrary shape
- Disadvantage: Sensitive to parameters

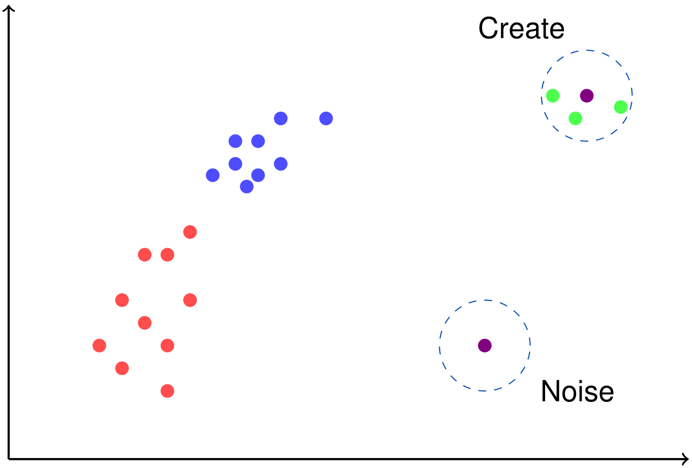
Incremental DBSCAN (Addition)



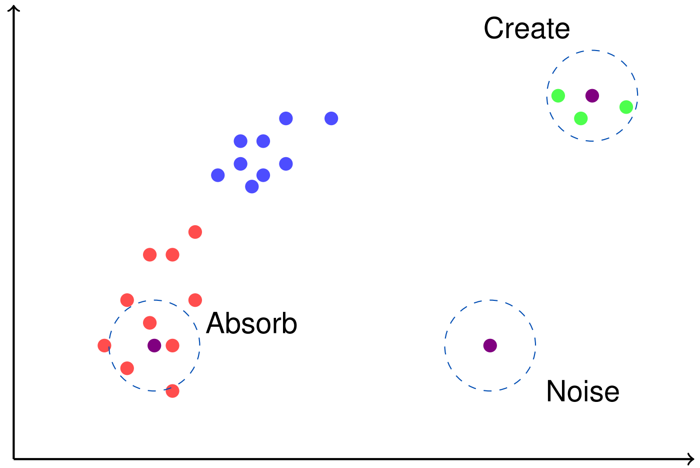
Incremental DBSCAN (Addition)



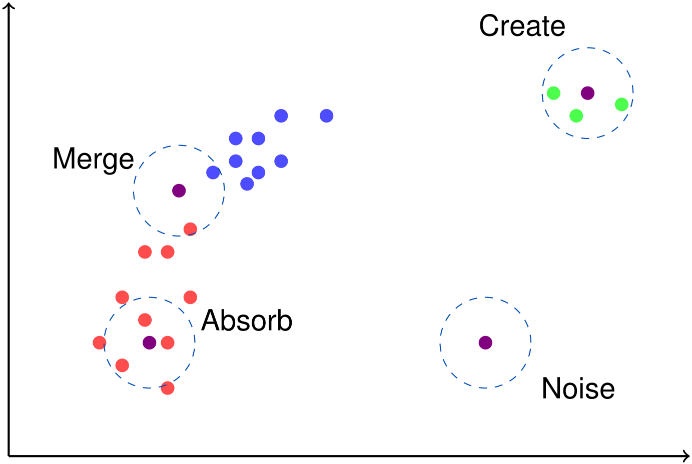
Incremental DBSCAN (Addition)



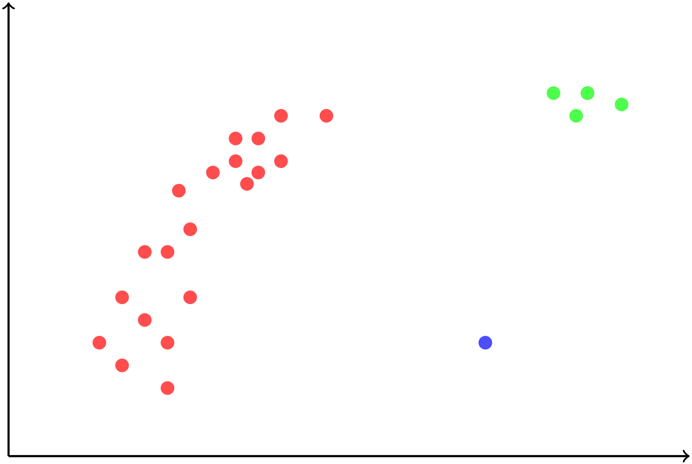
Incremental DBSCAN (Addition)



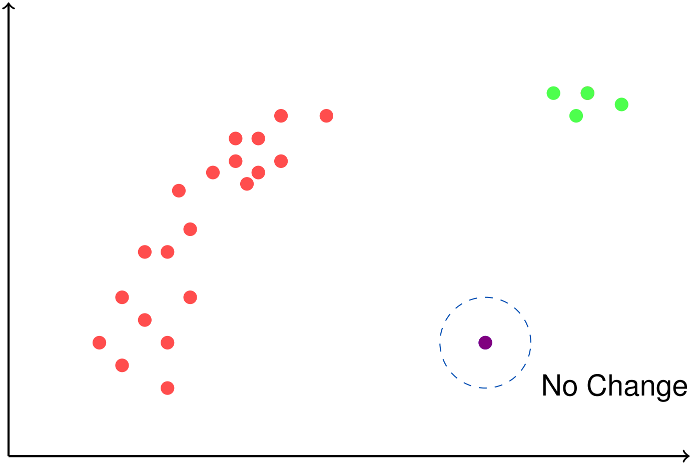
Incremental DBSCAN (Addition)



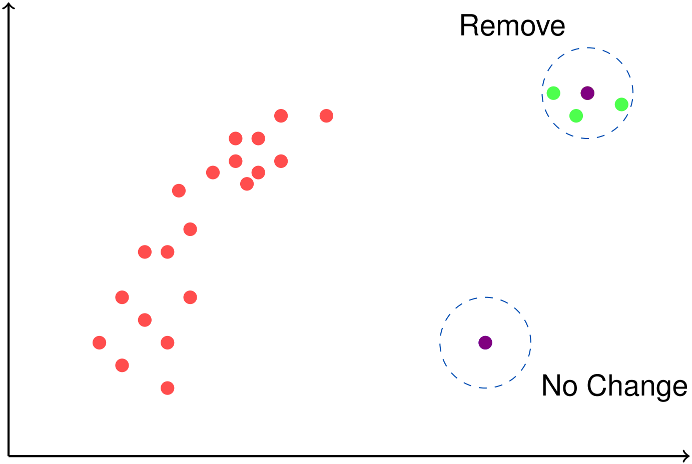
Incremental DBSCAN (Deletion)



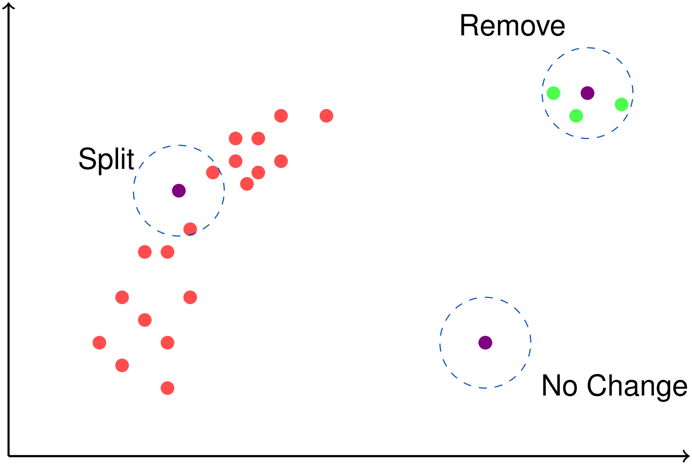
Incremental DBSCAN (Deletion)



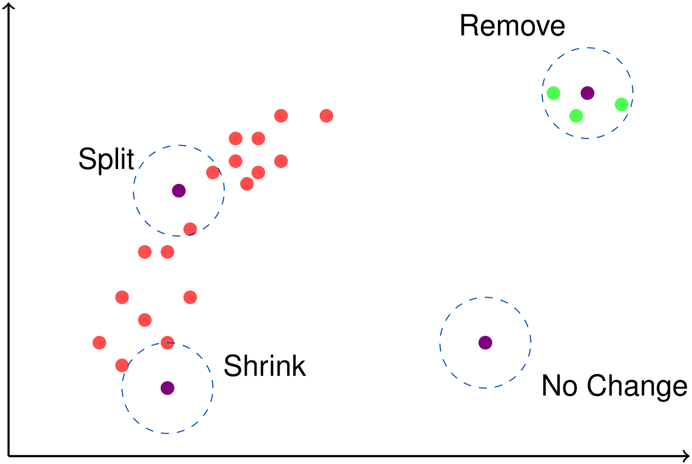
Incremental DBSCAN (Deletion)



Incremental DBSCAN (Deletion)



Incremental DBSCAN (Deletion)



Incremental DBSCAN

- Insertion and deletion are treated separately
- Based on change in density in affected region, clusters are updated.
- Update cost is proportional to number of points in affected region that is high
- You may be doing redundant operations

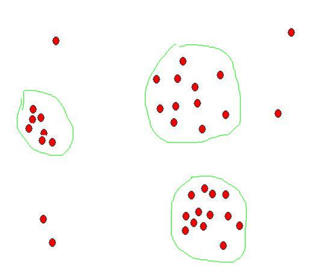
Differ update for some time

Assume periodic arrival of updates.

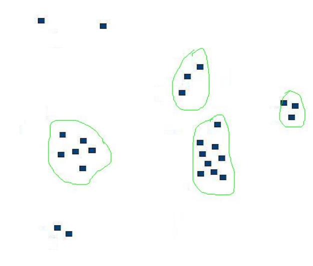
- Cluster new data
- Merge it with previous clusters (it is easy to see the density change)

Incremental DBSCAN

Initial Database

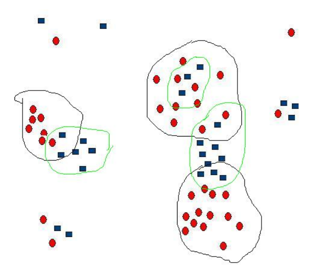


New Date

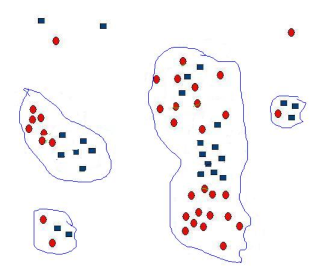


- Region based merging is applied

Incremental DBSCAN



Overlapping of clusters made from original and the new data points looks as



- Point p is in set of intersection I' if $\exists p' \in D$ such that p and p' are neighbor
- It is necessary an sufficient to process all $p \in I'$
- Efficiently compute I'. How?

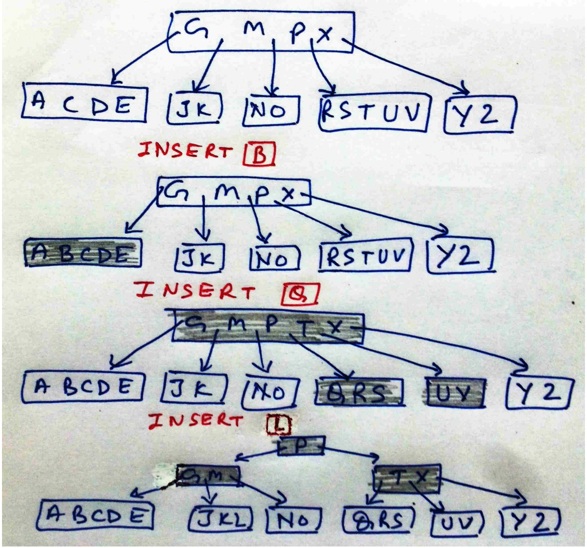
R-tree an efficient date structure

- R-tree provides an approach to index multidimensional spatial data
- Locating a nearest object to a current location is easy by using R-tree
- Or finding all objects in vicinity
- Uses a minimum bounding rectangle (MBR)
- It is a smallest rectangle that always contains the specified object
- Each node in the index contains its children
- Leaves of the tree points the actual objects
- Tree is height-balanced so height is $O(\log n)$
- R-Tree node usually corresponds to database points
- Similar to B-Tree

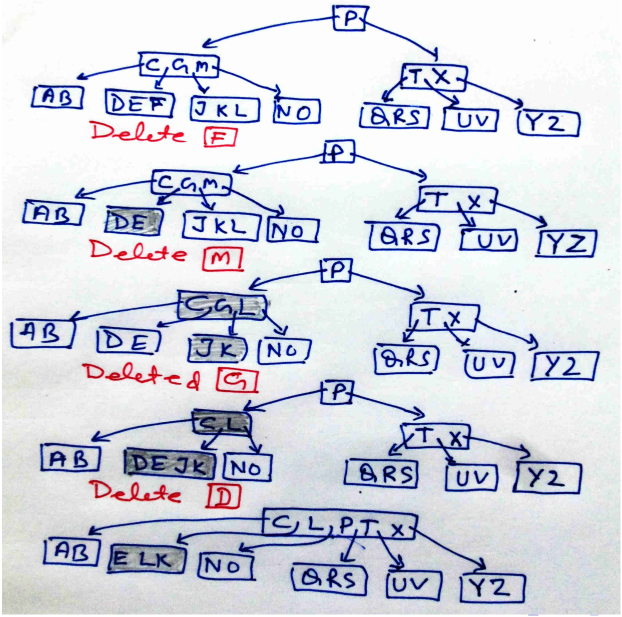
Remember B-tree

- B-Tree is a rooted tree. Every node x has $n(x)$ number of keys with $key_1[x] \leq key_2[x] \leq ... \leq key_{n[x]}[x]$ and $leaf[x] = TRUE$ if it is a leaf node
- Internal node also contains $n(x) + 1$ pointers $c_1[x], c_2[x], ..., c_{n[x]+1}[x]$
- For chosen $k > 1$, three exists at least $k - 1$ and at most $2k - 1$ keys at every node except root
- Height of tree $h \leq \log_t \frac{n+1}{2}$

B-tree (insertion)



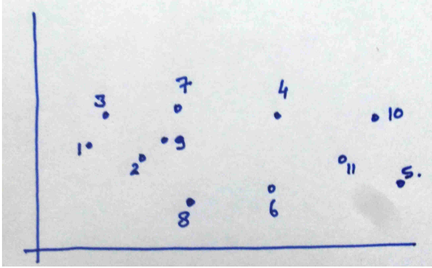
B-tree (deletion)

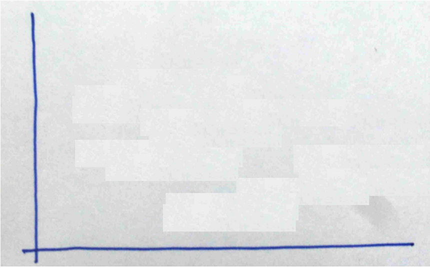


R-tree at Work

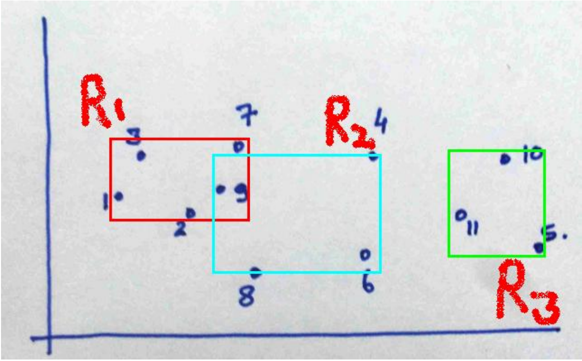
Consider arrival of

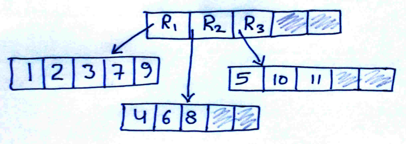
- P_1, P_2, P_3, P_4, P_5
- P_6 (split and region formation)
- P_7 (R1 expands)
- P_8 (R2 expands)
- P_9, P_{10}
- P_{11} (Split in R2)





R-tree at Work

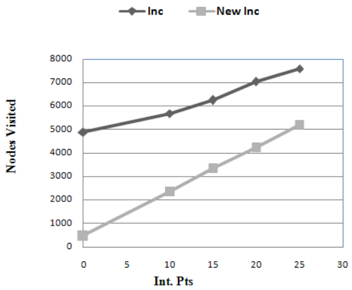


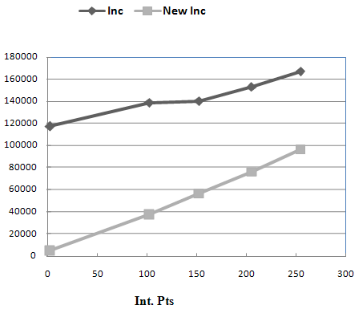


Incremental DBSCAN

- Find overlap region between R and R' with two trees holding objects from D and D'
- This would contains all the points under two nodes which are ϵ distance apart
- Is constructed by taking the intersection of ϵ-expanded MBRs pf the two nodes and the overlap region of the parent nodes
- Relative improvement is high if there is less overlap

http://cs.joensuu.fi/sipu/datasets/





Thank You!

Thank you very much for your attention!

Queries ?