# CLUSTERING

CURE and ROCK

## CURE

- CURE ≡ Clustering Using REpresentatives
- Problem with many clustering methods
  - ➢ favor clusters with spherical “or convex” shapes and similar sizes
  - ➢ sensitive to outliers
- CURE can find clusters of any shape and size and it detects outliers
- CURE is a mixed partitional and hierarchical method

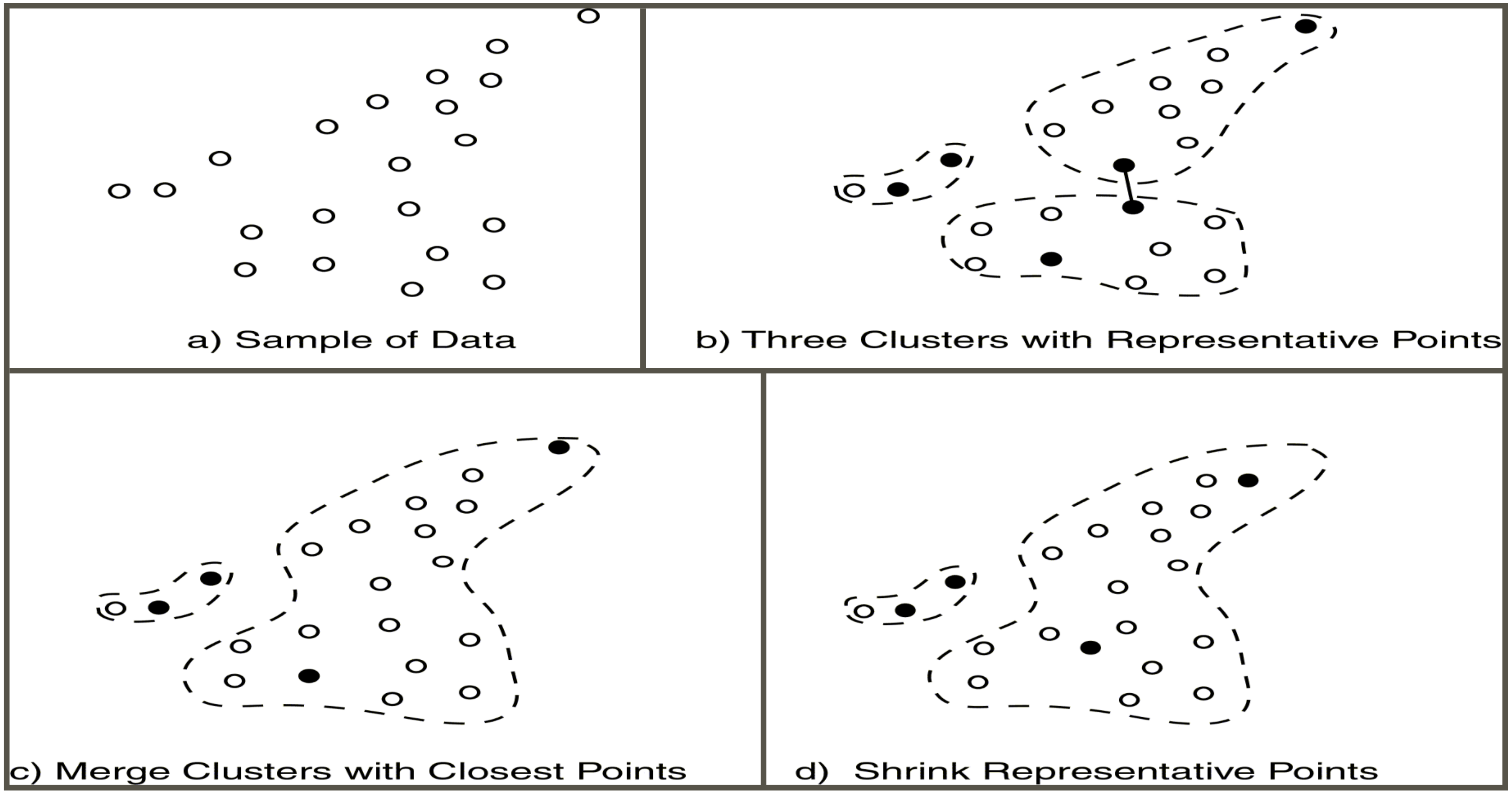
## Idea

- Several points are used to represent a cluster
- This is more descriptive of clusters of arbitrary shapes
- A set of c well-scattered points from the cluster are used as representatives
  - 1st representative point is farthest from the centroid
  - other scattered points are chosen so that their distance from previously chosen scattered points is maximal

## Idea (cont.)

- The well-scattered representative points are then shrunk towards the centroid by a shrinking factor, $\alpha$
  - When $\alpha$ is 1, all representative points are shrunk to one point, the cluster's centroid
  - When $\alpha$ is $0$, no shrinking is done
  - a value between $0.2$ to $0.7$ is usually used
- Shrinking the scattered points toward the centroid reduces the effects of outliers
- In the hierarchical step, the two closets clusters are merged
  - distance between two clusters is the closest distance between a pair of representatives
- Every time two clusters are merged their representatives are re-calculated
- CURE uses a random sample and partitioning to handle large datasets

## CURE Approach


a) Sample of Data
b) Three Clusters with Representative Points
c) Merge Clusters with Closest Points
d) Shrink Representative Points

# Outline of CURE

1. Draw a random sample, S, of objects
   - sample size ~ $2.5\%$ of the size of the dataset
2. Partition the sample S into a set of partitions
   - number of partitions greater than two or three times # of clusters
3. Partially cluster each partition
   - a method like nearest neighbor can be used
   - stop when number of clusters ~ one third number of objects in partition
4. Eliminate outliers
   - outliers are assumed to be small clusters with 1 or 2 objects

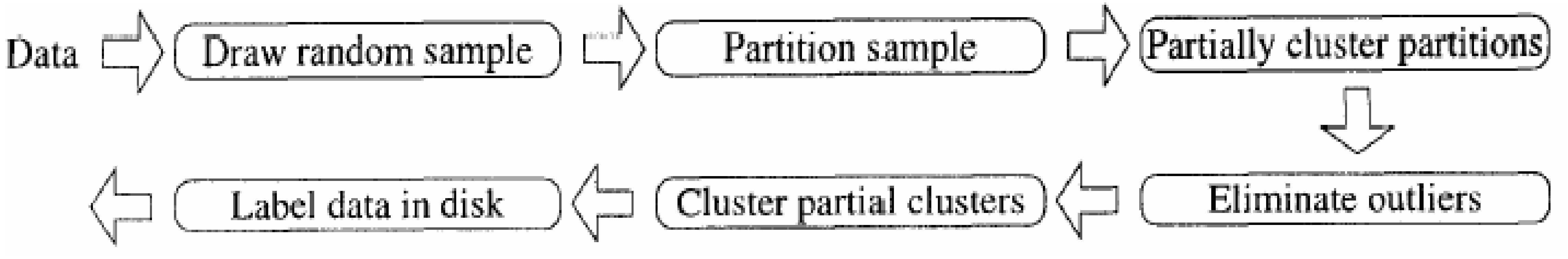
## Outline of CURE (cont.)

5. Cluster the partial clusters
   - a hierarchical method is used
   - two clusters k1 and k2 are merged if they are closest to each other
     dist(k1, k2) = min(p, q) where p ∈ k1.rep, q ∈ k2.rep
   - find representative points for the new cluster
   - shrink representatives by a shrinking factor $\alpha, 0 \leq \alpha \leq 1$
   - stop when k clusters are found

6. Cluster data on disk
   - Assign a point to the cluster with the closest representative

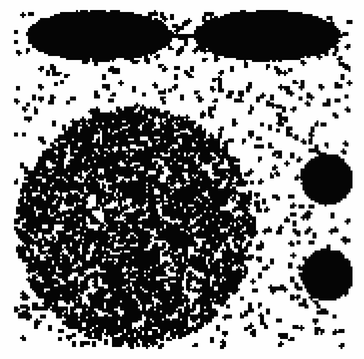
7. Remove small clusters as outliers

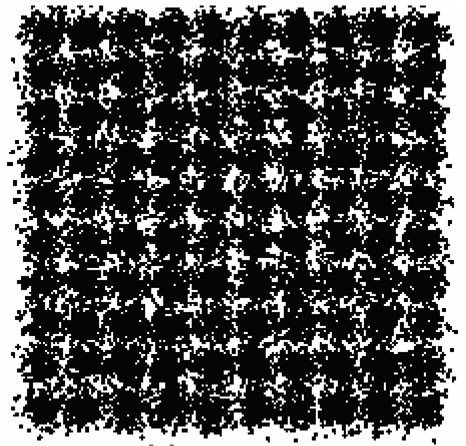
# Overview of CURE



## Experimental Results

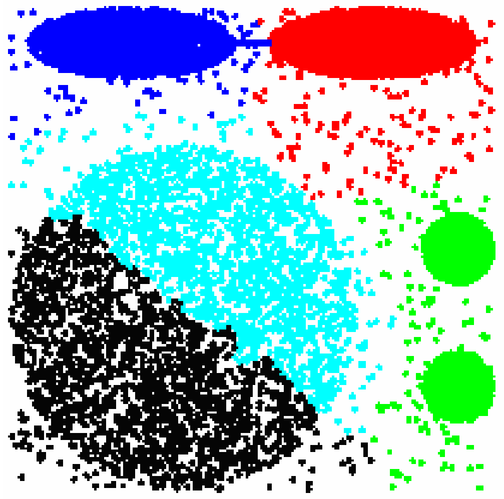
- Compared three algorithms on two datasets
  - ➢CURE
  - ➢BIRCH
  - ➢MST (Minimum Spanning Tree)
- Dataset 1 consists of one big circle, two small circles, and two ellipsoids
- Dataset 2 consists of 100 clusters with centers arranged in a grid pattern and points in each cluster following a normal distribution with mean at the cluster center

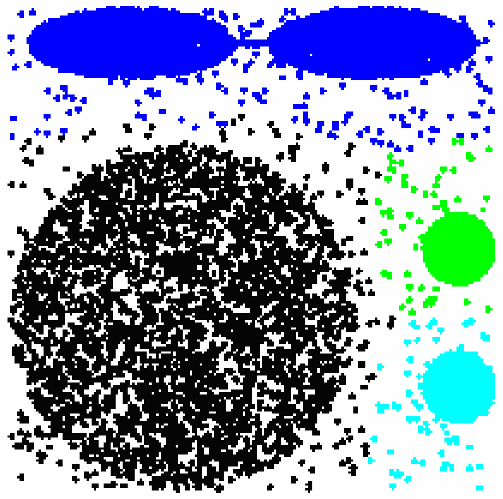

(a) Data set 1

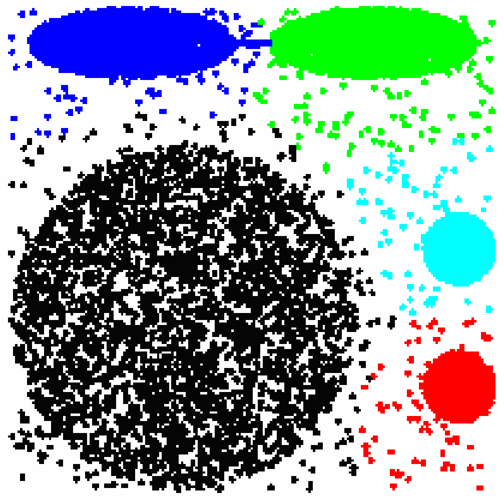

(b) Data set 2

## Experimental Results – Quality of Clustering

- BIRCH split one cluster and merged two clusters
- MST merges the two ellipsoids (long chain effect)
- CURE successfully discovered the clusters in Data set 1


(a) BIRCH

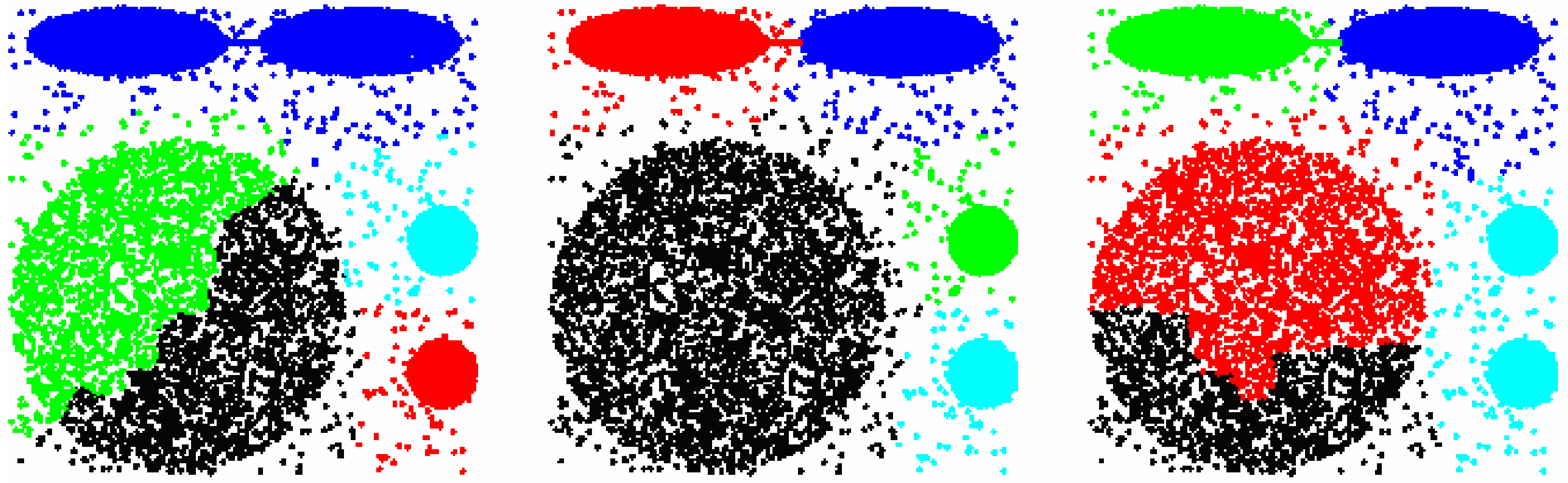

(b) MST METHOD


(c) CURE

# Experimental Results – Sensitivity to Parameters

Shrinking factor $\alpha$:

- $0.2 - 0.7$ is a good range of values for $\alpha$.

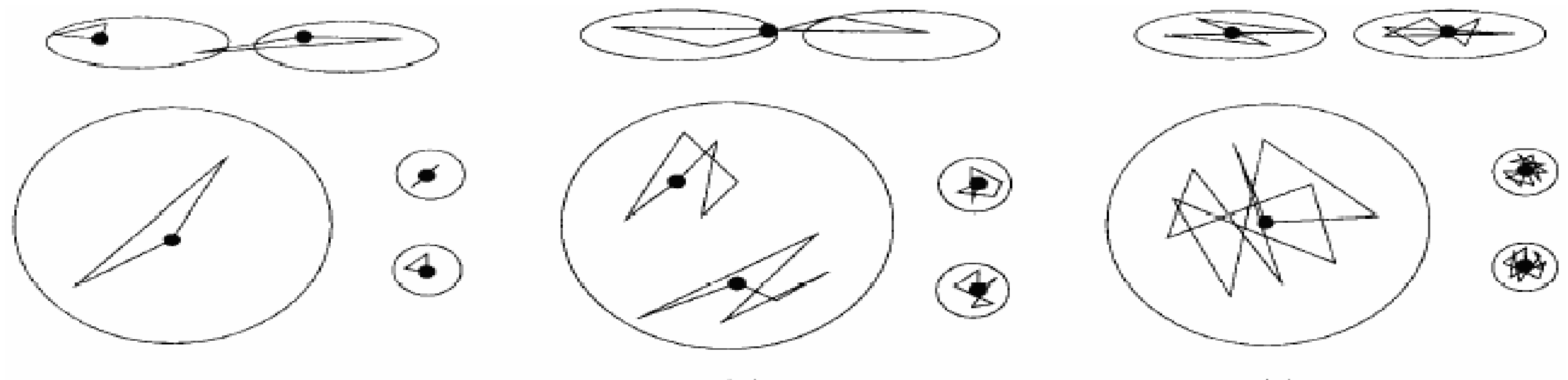


(a) $\alpha = 0.1$ (b) $\alpha = 0.2 - 0.7$ (c) $\alpha = 0.8 - 0.9$

## Experimental Results – Sensitivity to Parameters (cont.)

Number of Representative Points c:

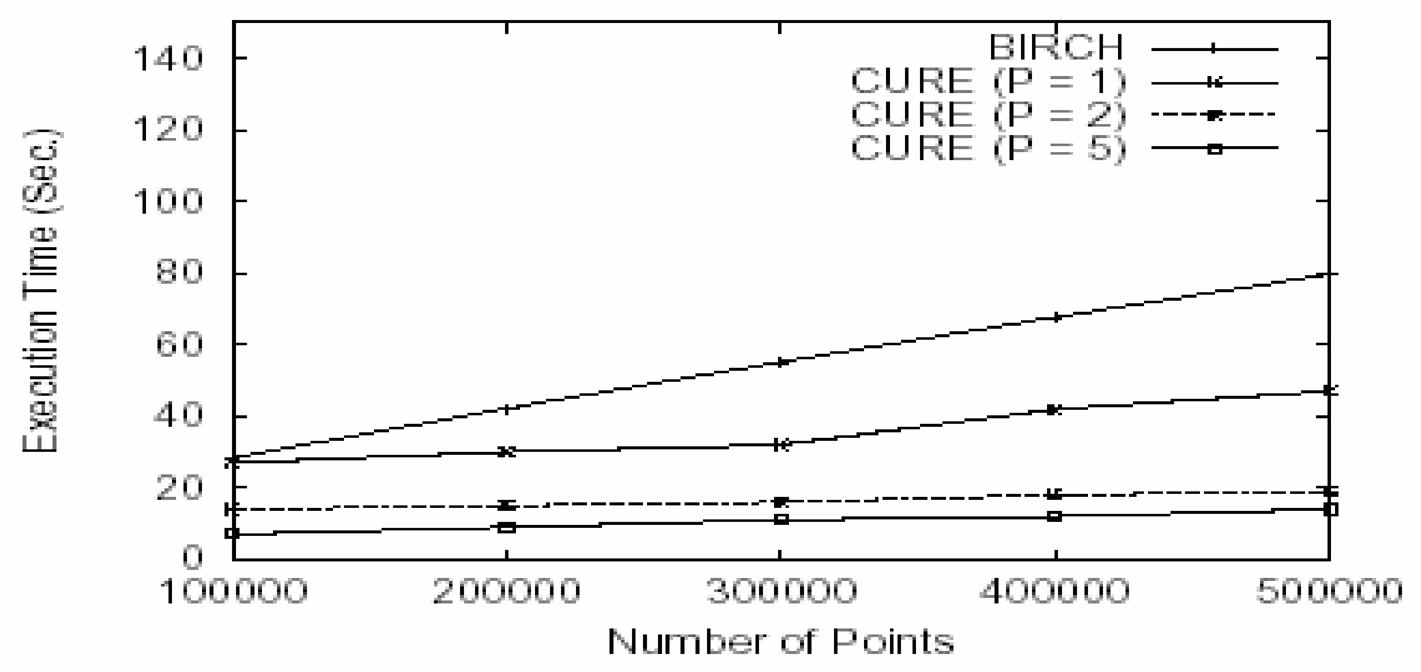
- For smaller values of c, the quality of clustering suffered.
- However, for values of c greater than 10, CURE always found right clusters.



(a) $c = 2$ (b) $c = 5$ (c) $c = 10$

## Experimental Results – Comparison of Execution time to BIRCH

Run both BIRCH and CURE on Data set 2



## Conclusion

- CURE can adjust well to clusters having non-spherical shapes and wide variances in size
- CURE can handle large databases efficiently (one scan)

# Clustering with Categorical Attributes

- Algorithms suited for numeric data, do not work well for categorical attributes
- Viewing attribute values as $0/1$ pairs or doing simple numeric mapping does not work (next slide)
- Example next page shows that using centroid-based method and Euclidean distance for clustering fails
- There are many clustering algorithms for categorical attributes including: STIRR, ROCK, CACTUS, and c-modes
- We'll talk about the ROCK algorithms

## Shortcomings of Traditional Methods on Categorical Data

- Consider a market basket dataset containing $4$ transactions over the set of items {a, b, c, d, e, f}

t1 = {a, b, c, e }
t2 = { b, c, d, e }
t3 = {a, d }
t4 = { f}

- Represent transactions as $0/1$ vectors

- Representation: a b c d e f

t1 = $(1, 1, 1, 0, 1, 0)$
t2 = $(0, 1, 1, 1, 1, 0)$
t3 = $(1, 0, 0, 1, 0, 0)$
t4 = $(0, 0, 0, 0, 0, 1)$

# Shortcomings of Traditional Methods on Categorical Data (cont)

- $$\begin{array}{lcl} & & \text{a b c d e f} \\ t_1 & = & (1,1,1,0,1,0) \\ t_2 & = & (0,1,1,1,1,0) \\ t_3 & = & (1,0,0,1,0,0) \\ t_4 & = & (0,0,0,0,0,1) \end{array}$$
- Use centroid-based hierarchical clustering and Euclidean distance to measure distances:
- t1, t2 are closest, so merge in cluster, K1
- centroid of K1 = $(0.5, 1, 1, 0.5, 1, 0)$
- d(t3, K1) = $\sqrt{3.5}$, d(t4, k1) = $\sqrt{4.5}$, d(t3, t4) = $\sqrt{3}$
- next, algorithm merges t3 and t4
- this corresponds to merging transactions t3 = {a, d} and t4 = {f}
- so, this method fails on categorical data

| | t1 | t2 | t3 | t4 |
|---|---|---|---|---|
| t1 | $0$ | $\sqrt{2}$ | $2$ | $\sqrt{5}$ |
| t2 | | $0$ | $2$ | $\sqrt{5}$ |
| t3 | | | $0$ | $\sqrt{3}$ |
| t4 | | | | $0$ |

## ROCK Algorithm

- **ROCK** ≡ **RO**bust **C**lustering using lin**K**s
- It is an agglomerative hierarchical clustering method for categorical attributes
- It uses the idea of neighbors and number of links between two items for similarities
- Two objects are neighbors if they are similar enough to each other
- Link for pair of objects is the number of common neighbors.

# Neighbors

- Def: A pair of tuples $t_i$ and $t_j$ are neighbors if their similarity is at least some threshold value, $\theta$

  i.e., $t_i$ and $t_j$ are neighbors if $\text{sim}(t_i\, ,\, t_j\,) \geq \theta$

- The similarity measure used in ROCK is based on Jaccard coefficient and is defined as follows:

$$\text{sim}(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|}$$

- Example: Find the similarity between $t_1 = \{a, b, c, e\}$ and $t_2 = \{b, c, d, e\}$

$$\text{sim}(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|} = \frac{|\{a,b,c,e\} \cap \{b,c,d,e\}|}{|\{a,b,c,e\} \cup \{b,c,d,e\}|} = \frac{|\{b, c, e\}|}{|\{a,b,c,d,e\}|} = \frac{3}{5} = 0.6$$

## Links

- Def: The number of links between two objects $t_i$ and $t_j$, $\text{link}(t_i, t_j)$, is defined as the number of common neighbors they have
- Note that a point is considered as a neighbor of itself

# Example

- Consider an information retrieval system where documents have the keywords

  {Book, Water, Sun, Sand, Swim, Read}

- Given the four documents

  **d1** = {Book}, **d2** = {Water, Sun, Sand, Swim}

  **d3 =** {Water, Sun, Swim, Read}, **d4** = {Read, Sand}

- similarity matrix using sim$(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|}$

|  | d1 | d2 | d3 | d4 |
|---|---|---|---|---|
| d1 | 1 | 0 | 0 | 0 |
| d2 |  | 1 | 0.6 | 0.2 |
| d3 |  |  | 1 | 0.2 |
| d4 |  |  |  | 1 |

- Using threshold $\theta = 0.2$, neighbors for each element are

| document | neighbors |
|---|---|
| d1 | d1 |
| d2 | d2, d3, d4 |
| d3 | d2, d3, d4 |
| d4 | d2, d3, d4 |

- Table showing link$(di, d_j)$

|  | d1 | d2 | d3 | d4 |
|---|---|---|---|---|
| d1 | 1 | 0 | 0 | 0 |
| d2 |  | 3 | 3 | 3 |
| d3 |  |  | 3 | 3 |
| d4 |  |  |  | 3 |

- Resulting Clusters after applying ROCK {{d1}, {d2, d3, d4}}

## Outline of the ROCK Algorithm

- Draw a random sample, S, of the data
- Perform clustering on the sample using the link agglomerative hierarchical clustering
  - merge clusters with the largest number of links until k clusters are formed
  - algorithm uses goodness measure $g(C_i, C_j)$ that takes cluster size into consideration
- Cluster data on disk
  - an object is assigned to the cluster with which it has the highest number of links

## Remarks

- For a pair of clusters $C_i, C_j$, link($C_i, C_j$) **is the number of mutual links between $\boldsymbol{C_i}$ and $\boldsymbol{C_j}$**

  i.e., link($C_i, C_j$) $= \sum_{t_i \in Ci, tj \in Cj}$ link$(t_i, t_j)$

- The goodness measure g ($C_i, C_j$) is directly proportional to link($C_i, C_j$) and takes into consideration the size of the clusters
- A pair of clusters with maximum g is the best pair to merge