Cure

Cluster Using REpresentative i.e. CURE is very efficient data clustering algorithm for specifically large databases.
CURE is robust to outliers.

Traditional clustering algorithm :

In traditional clustering, it selects for any one point and it is only point considered as a cluster i.e. clusters centroid
approach.
Points in a cluster appear close to each other compared to other data points of any other clusters. It works in eclipse
shape in better way.
Drawback of traditional clustering algorithm is all-points approach makes algorithm highly sensitive to outliers and a
minute change in position of data points.
Cluster centroid and all points approach not work on arbitrary shape.

CURE Algorithm

CURE algorithm works better in spherical as well as non-spherical clusters.
– CURE : An efficient clustering algorithm for large database : sudipto Guha, Rajeev Rastogi, Kyuseok Shim.
– It prefers a set of points which are scattered as representative cluster than all-points or centroid approach.
– CURE uses random sampling and partitioning to speed up clustering.

Overview of CURE (Cluster Using REpresentative)

Data
↓
Make random sample
↓
Make partitioning of sample
↓
Partially cluster partitions
↓
Eliminate outliers
↓
Cluster partial clusters
↓
Label data in disk

Hierarchical Clustering Algorithm

– A centroid-based point 'c' is chosen. All remaining scattered points are just at a fraction distance of α to get shrunk
towards centroid.
– Such multiple scattered points help to discover in non spherical cluster i.e. elongated cluster.
– Hierarchical clustering algorithm uses such space which is linear to input size n.
– Worst-case time complexity is O (n_2 long n) and it may reduce to O(n_2) for lower dimensions.

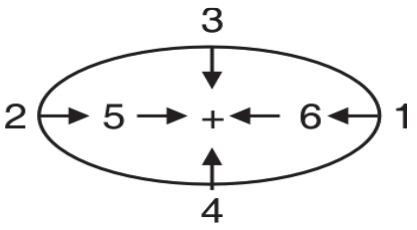
CURE algorithm : CURE cluster procedure

– It is similar to hierarchical clustering approach. But it use sample point variant as cluster representative rather than
every point in the cluster.

– First set a target sample number C. Then we try to select C well scattered sample points from cluster.
– The chosen scattered points are shrunk towards the centroid in a fraction of α where$0 \leq \alpha \leq 1$.

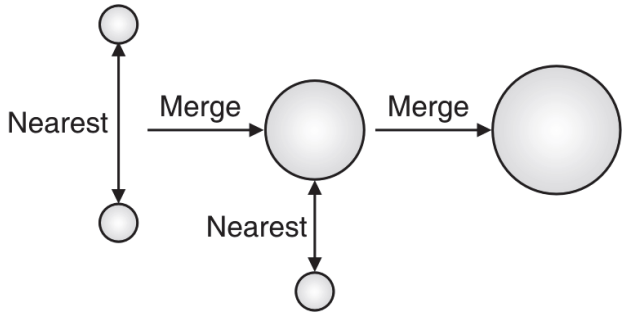
Fig.

– These points are used as representative of clusters and will be used as point in dmin cluster merging approach.
– After each merging, C sample points will be selected from original representative of previous clusters to represent
new cluster.



These points are used as representative of clusters and will be used as point in dmin cluster merging approach.
– After each merging, C sample points will be selected from original representative of previous clusters to represent
new cluster.

Cluster merging will be stopped until target K cluster is found.



Random Sampling and Partitioning Sample

To reduce size of input to CURE's clustering algorithm random sampling is used in case of large data sets.
– Good clusters can be obtained by moderate size random samples, it provides tradeoff between efficiency and accuracy.
– Portioning sample reduces time required for execution because before final cluster made each partition get clustered whenever it is in pre-clustered data format at eliminated outliers.

Eliminate Outlier's and Data Labelling

Outliers points are generally less than number in cluster.
– As random sample gets clustered, multiple representative points from each cluster are labelled with data set remainders.
– Clustering based on scattered point i.e. CURE approach found most efficient compared to centroid or all-points approach of traditional clustering algorithm.

Pseudo function of CURE (clustering algorithm)

```
Procedure cluster (s, k)
Begin
T : = build – kd – tree (s)
Q : = build – heap (s)
While size (Q) > k
do {
u : = extract – min (Q)
v : = u – closest
delete (Q, v)
w : = merge (u, v)
delete – rep (T, u) ;
delete – rep (T, v) ;
insert – rep (T, w) ;
w – closest : = x
```

for each x □ Q
do {
if dist (w, x) < dist (w, w – closest)
w – closest : = x
if x – closest is either u or v {
if dist (x, x – closest) < dist (x, w)
x – closest : = closest – cluster
(T, x, dist (x, w))
else
x – closest : = w
relocate (Q, x)
}
else if dist (x, x–closest) > dist (x, w){
x – closest : = w
relocate (Q, x)
}
}
insert (Q, w)}

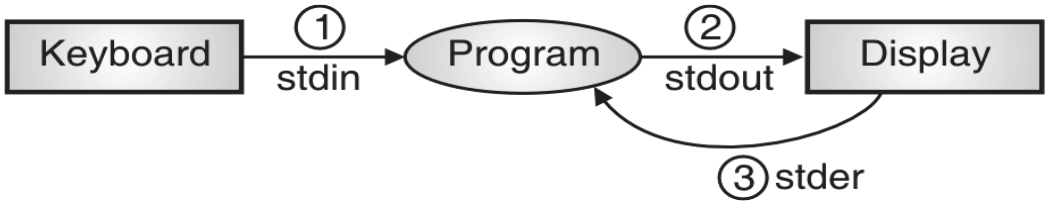
Procedure for merging clusters

Procedure merge (u, v)
being
w : = u U v
w. mean : = |u| u.mean + |v| v-mean/|u| + |v|
'tmpset : = ϕ
For i : = 1 to c do {
maxDist : = 0
for each point p in cluster w do {
if i = 1
min Dist : = dist (p, w, mean)
else
min Dist : = min {dist (p, q) : q € tmpset}
if (min Dist > max Dist)
{ max Dist : = min Dist
Max point : = P
}
}
tmpset : = tmpset U {maxpoint}
}

For each point P in tempest do
w – rep : = w_rep □ {p + □ □ (w-mean-p)}
return w
end

Stream Computing

– Stream computing is useful in real time system like count of items placed on a conveyor belt.
– 1BM announced stream computing system in 2007, which runs 800 microprocessors and it enables to software applications to get split to task and rearrange data into answer.
– AT1 technologies derives stream computing with Graphical Processors (GPUs) working with high performance with low latency CPU to resolve computational issues.
– AT1 preferred stream computing to run application on GPU instead of CPU



A Stream - Clustering Algorithm

– BDMO Algorithm has complex structures and it is designed in approach to give guaranteed performance even in worst
case.
– BDMO designed by B. Bahcock, M. Datar, R. Motwani and L. OCallaghan.

Details of BDMO algorithm

(i) Stream of data are initially partitioned and later summarized with help of bucket size and bucket is a power of two.
(ii) Bucket size has few restrictions size of buckets are one or two of each size within a limit. Required bucket may start
with sized or twice to previous for example bucket size required are 3, 6, 12, 24, 48 and so on.
(iii) Bucket size are restrained in some scenario, buckets mostly O (log N).
(iv) Bucket consists with contents like size, timestamp, number of points in cluster, centriod etc.
Few well – known algorithm for data stream clustering are :
(a) Small – Spaces algorithm (b) BIRCH
(c) COBWEB (d) C2ICM

Initializing and Merging Buckets

A small size 'p' is chosen for bucket where p is power of 2. Timestamp of this bucket belongs to a timestamp of most recent points of bucket.
– Clustering of these points done by specific strategy. Method preferred for clustering at initial stage provide the centriod or clustroids, it becomes record for each cluster.
Let,
* 'p' be smallest bucket size.
* Every p point, creates a new bucket, where bucket is time stamped along with cluster points.
* Any bucket older than N is dropped

* If number of buckets are 3 of size p

p → merge oldest two

– Then propagated merge may be like (2_p, 4_p, …).

– While merging buckets a new bucked created by review of sequence of buckets.

– If any bucket with more timestamp than N time unit prior to current time, at such scenario nothing will be in window of the bucket such bucket will be dropped.

– If we created p bucket then two of three oldest bucket will get merged. The newly merged bucket size nearly z_p, as we needed to merge buckets with increasing sizes.

– To merge two consecutive buckets we need size of bucket twice than size of 2 buckets going to merge. Timestamp of newly merged bucket is most recent timestamp from 2 consecutive buckets. By computing few parameters decision of cluster merging is taken.

– Let, k-means Euclidean. A cluster represent with number of points (n) and centriod (c).

Put p = k, or larger – k-means clustering while creating bucket

To merge, $n = n_1 + n_2$, $c = n_1c_1 + n_2c_2 / n_1 + n_2$

– Let, a non Euclidean, a cluster represented using clusteroid and CSD. To choose new clusteroid while merging, k-points furthest are selected from clusteroids.

$$CSD_m(P) = CSD_1(P) + N_2(d_2(P, c_1) + d_2(c_1, c_2)) + CSD_2(c_2)$$

Answering Queries

– Given m, choose the smallest set of bucket such that it covers the most recent m points. At most 2m points.

– Bucket construction and solution generation are the two steps used for quarry rewriting in a shared – variable bucket algorithm, one of the efficient approaches for answering queries.