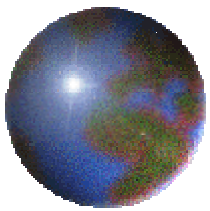
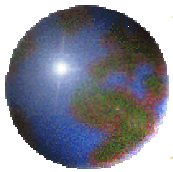


An Optimized Algorithm for Fast Projected Clustering (OPTICLUS)

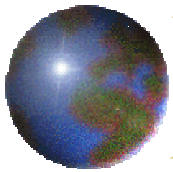
Joe Zhou, Wendy Wang

Dec. 10, 2003



Outline

- Introduction
- Related work ---PROCLUS
- Algorithm Analysis
- Empirical Results
- Conclusions & Future Work



Clustering in High Dimensional Space

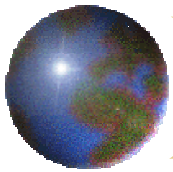
- Definition

  objects within a cluster are similar between themselves and dissimilar to objects of other clusters

- Challenges in High Dimension Space

  Dimensionality Curse

$$\lim_{d \to \infty} \frac{D\ \max_d - D\ \min_d}{D\ \min_d} = 0$$



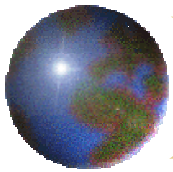
Projected Clustering

- Projected Cluster

  subset – subspace pair (C, D)

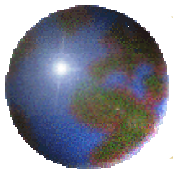
  data points in C are closely clustered in dimension subset D

- Partitioning Clustering Algorithms
  - PROCLUS
  - ORCLUS



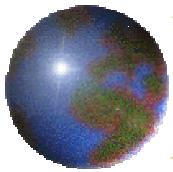
Related work----PROCLUS

- *Input:* number of clusters k and average subspace dimension l
- *Output:* k good medoids with subspaces
- Hill climbing technique
  - Take a set of k medoids
  - Find subspace for each medoid using its locality
  - Given a set of medoids and their associated subspace, partition data set
  - Change the bad medoid



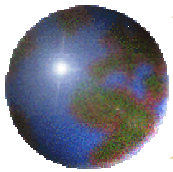
PROCLUS Analysis

- Initialization Phase
  - Random select A*K data points
  - greedy select B*K piercing medoid candidates
  - random select K medoids from B*K candidates
- Iterative Phase
  - Find subspaces for each medoid in its associated locality
  - Assign data records to each medoid to form clusters
  - Evaluate and change the bad medoid
- Refinement Phase
  - Refine clusters and its associated subspaces with one data pass



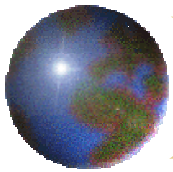
Our Contribution

- Analyze and find problems in existing projected clustering algorithms
- Tackle the problem and propose a new algorithm based on PROCLUS
- Implement the algorithm, called OPTICLUS, as well as PROCLUS and test on the generated synthetic data



Synthetic Data Generation

Data Parameter	Values used
Number of records($\mathcal{N}$)	100,000
Dataset dimensionality(d)	20-100
Number of clusters(k)	5
Cluster dimensionality	20-80% of d
Cluster size	3-70% of $\mathcal{N}$
Local variance s_{ij} of relevant attributes	0-2% of the domain
Artificial outliers (o)	5%



Empirical Results--- Case 1:

- The input clusters are of the same size
- Good result of OPTOCLUS, compared with that of PROCLUS with the correct parameter l.
- Bad result of PROCLUS with inaccurate paramter l.

Input Clusters

Input	Dimensions	Points
1	0 1 2 4 5 7 9 14 18	20000
2	1 2 6 7 12 13 15 16 18	20000
3	1 2 3 4 7 8 11 13 17 19	20000
4	1 2 3 4 5 6 8 9 10 13 14	20000
5	1 2 3 4 9 10 13 15 17 18 19	20000

Output of PROCLUS with $l = 11$

Found	Dimensions	Points
A	1 2 3 4 7 8 11 13 17 19	21954
B	1 2 6 7 12 13 15 16 18	20549
C	1 2 3 4 9 10 13 15 17 18 19	20457
D	1 2 3 4 5 6 8 9 10 13 14	22024
E	0 1 2 4 5 7 9 14 18	20016

Note: Proclus find the correct clusters with correct dimension set given the accurate paramter l, average number of dimensions

Input Clusters

Input	Dimensions	Points
1	0 1 2 4 5 7 9 14 18	20000
2	1 2 6 7 12 13 15 16 18	20000
3	1 2 3 4 7 8 11 13 17 19	20000
4	1 2 3 4 5 6 8 9 10 13 14	20000
5	1 2 3 4 9 10 13 15 17 18 19	20000

Output Clusters of OPTICLUS

Found	Dimensions	Points
A	1 2 6 7 12 13 15 16 18	20536
B	1 2 3 4 9 10 13 15 17 18 19	20481
C	1 2 3 4 5 6 8 9 10 13 14	21946
D	0 1 2 4 5 7 9 14 18	20021
E	1 2 3 4 7 8 11 13 17 19	22016

Note: Optilcus also find the correct clusters with correct dimension set without any parameter like l

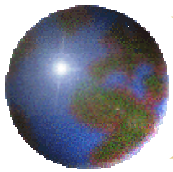
Input Clusters

Input	Dimensions	Points
1	0 1 2 4 5 7 9 14 18	20000
2	1 2 6 7 12 13 15 16 18	20000
3	1 2 3 4 7 8 11 13 17 19	20000
4	1 2 3 4 5 6 8 9 10 13 14	20000
5	1 2 3 4 9 10 13 15 17 18 19	20000

Output of PROCLUS with l = 7

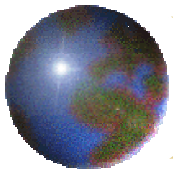
Found	Dimensions	Points
A	1 2 3 4 7 8 11 13	21613
B	1 2 6 7 12 13 15 16 18	20457
C	0 1 2 4 5 7 9 14 18	20013
D	3 4 8 13	21894
E	1 3 4 9 10	21023

Note: Proclus does not perform well when given the inaccurate parameter l .



Empirical Results--- Case 2:

- The input clusters vary much in size, from 6000 to 34000
- PROCLUS does not perform well given the accurate parameter /
- OPTICLUS find the correct clustering with right dimension sets



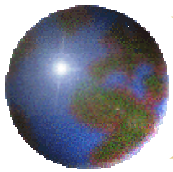
Input Clusters

Input	Dimensions	Points
1	0 1 2 4 5 7 9 14 18	6000
2	1 2 6 7 11 12 13 15 16 18	24400
3	1 2 3 4 7 8 9 11 13 16 17 18 19	34000
4	1 3 4 5 6 8 9 10 13 14 16	20000
5	1 2 4 5 6 9 10 13 15 17 18 19	15600

Output of PROCLUS

Found	Dimensions	Points
A	1 4 5 6 7 9 10 13 15	16211
B	1 2 6 7 11 12 13 15 16 18	24855
C	1 2 3 4 7 8 9 11 13 16 17 18 19	36062
D	0 1 2 3 5 7 11 13 14 15 18	6043
E	1 3 4 5 6 8 9 10 11 13 14 16	21829

Note: PROCLUS does not perform well when the cluster sizes are much different.



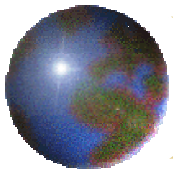
Input Clusters

Input	Dimensions	Points
1	0 1 2 4 5 7 9 14 18	6000
2	1 2 6 7 11 12 13 15 16 18	24400
3	1 2 3 4 7 8 9 11 13 16 17 18 19	34000
4	1 3 4 5 6 8 9 10 13 14 16	20000
5	1 2 4 5 6 9 10 13 15 17 18 19	15600

Output of OPTICLUS

Found	Dimensions	Points
A	1 2 6 7 11 12 13 15 16 18	24874
B	1 2 4 5 6 9 10 13 15 17 18 19	16030
C	1 3 4 5 6 8 9 10 13 14 16	22070
D	0 1 2 4 5 7 9 14 18	6032
E	1 2 3 4 7 8 9 11 13 16 17 18 19	35994

Note: Opticlus find the correct clusters with correct dimension set in the same case/.



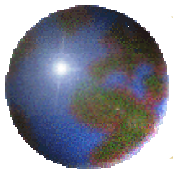
Future Work

- Finding low dimensional clusters

  OPTICLUS does not perform well for clusters which were related to relatively fewer attibutes, like less than 20% of the total attributes.

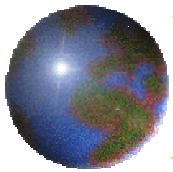
- Possible reasons
  - high dimension curse
  - outlier elimination in the initialization phase.

Questions & Discussions



Can Parameter l be kick out?

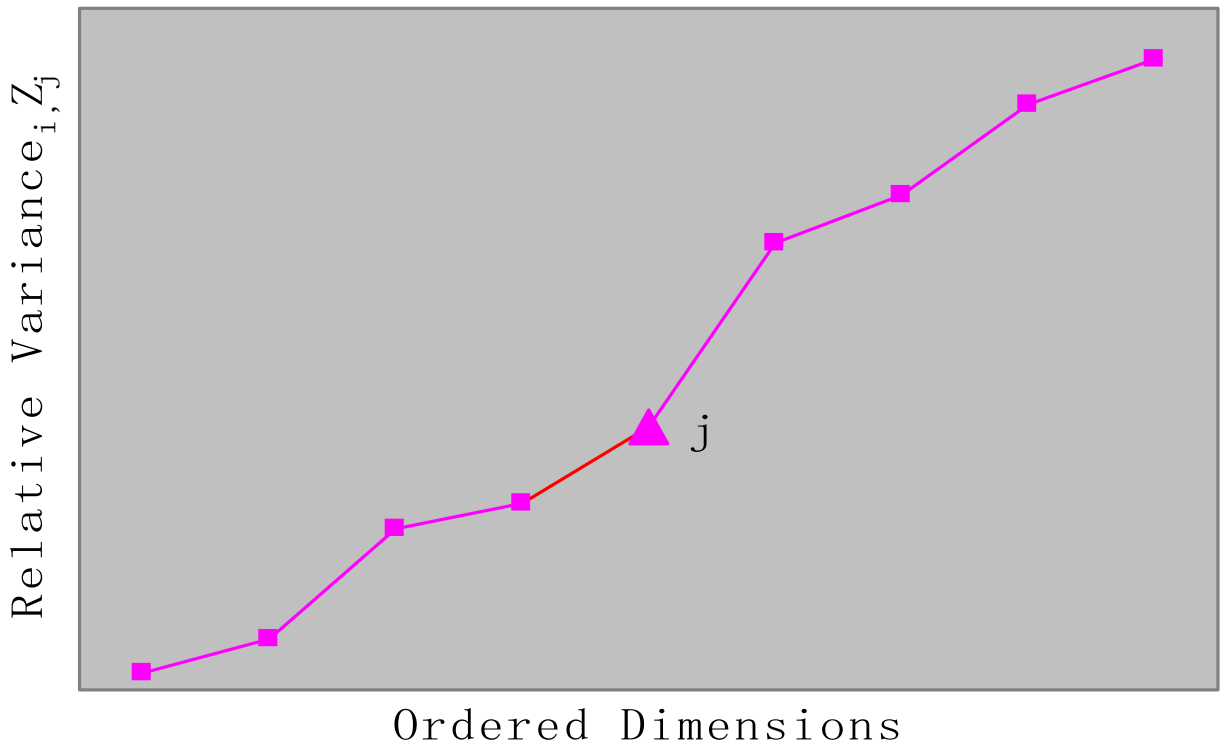
- *Chosen dimension have a much more smaller variance than irrelevant dimensions*
- Calculate "variance" on every dimension of each cluster
- Sort the variances and select l*k dimensions with least variance for all clusters
- User input parameter l is hardly known in real life application

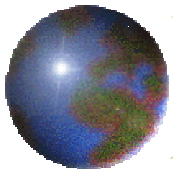


Automatic Dimension Finding

- Find the subset of dimensions independently for each cluster
- sort the variance in ascending order for each cluster
- Find the "sharp slope point" to get the set of dimensions
- Constraints: at least 2 and at most d – 2 dimensions are chosen for each cluster.

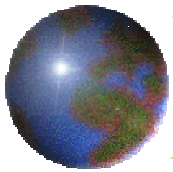
Automatic Dimension Finding



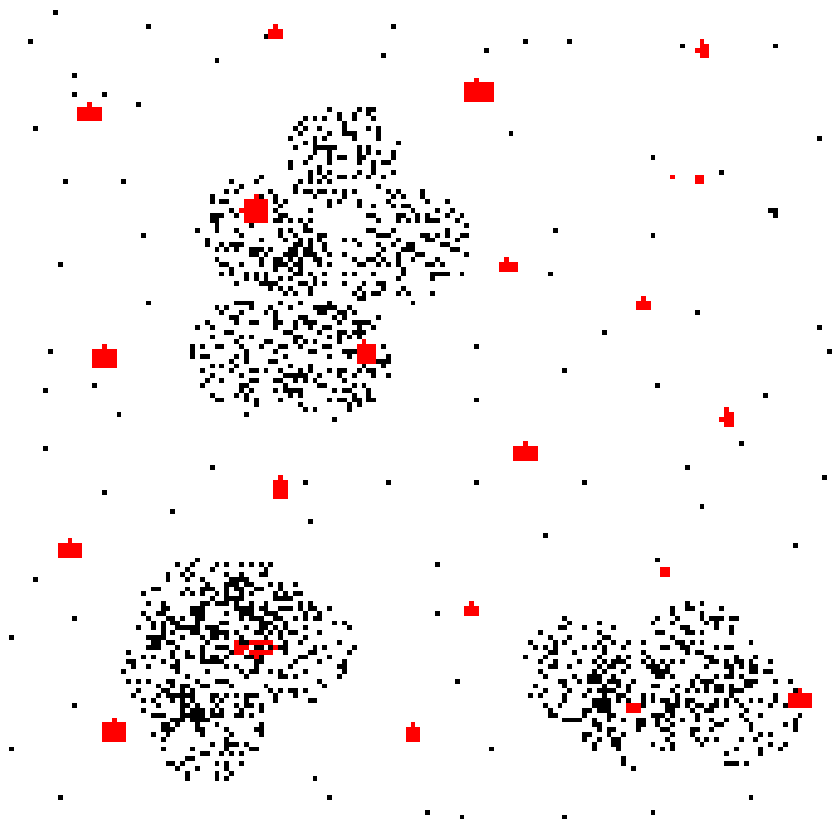


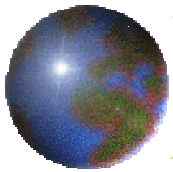
Piercing V.S. Outlier ?

- Piercing: A set of medoids is drawn from different cluster
- Use greedy algorithm to find the set of medoids which has the maximum distance in between
- Problem: the algorithm tends to select many outliers as medoids.



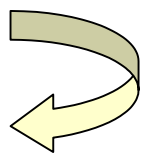
Piercing Medoid Set

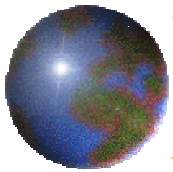




Outlier Elimination Technique

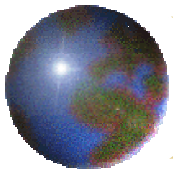
- Do a partition of the data set on the current medoids candidate
- Count the number of data points assigned to each medoid
- Eliminate the medoid that has least number of assigned data points
- The outliers are eliminated





What is a Bad Medoid ?

- Bad medoid is the medoid of the smallest cluster or clusters with size less than a specified value ----*PROCLUS*
- Effectively pick out outliers
- Ineffective when two medoids are falling in one big cluster



A New Criterion for "Bad"

- Assumption: The medoid that is relatively closer to centroid can form a good cluster
- The medoid farthest to its corresponding centroid is considered to be "bad"
- Facilitate finding clusters with different size

Medoid Optimization