A Scalable Parallel Subspace Clustering Algorithm for Massive Data Sets

Alok Choudhary

Harsha S Nagesh, Sanjay Goil
ECE Department
Northwestern University
Presentation Outline

- Clustering: Introduction
- Related Work
- Subspace Clustering
- Adaptive Grids
- pMAFIA: Design and Implementation
- Performance Results
- Conclusions
Clustering Algorithms: Issues to be Clustering

Discovery of *interesting* patterns in large multi-dimensional data sets.

- **What is the average credit for a particular income group (Financial Services)**
- **Areas with maximum collect calls (Telecommunications)**
- **Categorize stocks based on their movement (Investment Banking)**
- **Speech Analysis (Signal Processing)**
- **Analysis of satellite data (Scientific Data Management)**
Clustering Algorithms: Issues to be addressed

- Scalability with database size (out of core data sets)
- Scalability with the dimensionality of data
- Efficient Parallelization
- Unsupervised Clustering Algorithms
- Recognition of arbitrary shaped clusters
Related Work

• Partition based Clustering:
  – k-means, k-mediods, CLARANS (VLDB 94), BIRCH (SIGMOD 96), ...

• Hierarchical Clustering:
  – CURE (SIGMOD 98)

• Categorical Clustering:
  – Clustering of non-numerical data (e.g. automobile sales data: color, year, model, price, etc)
  – CACTUS (KDD 99), STIRR (VLDB 98)
Related Work

- Density and Grid Based Clustering:
  - Clusters are high density regions than its surroundings

WaveCluster (VLDB 98), DBSCAN, CLIQUE (SIGMOD 98), pMAFIA...

Grid Sizing is very critical!
Subspace Clustering

- Clusters in subspace of Multi-dimensional space.
- Subspaces is exponential in data dimensionality.

- **CLIQUE (SIGMOD 98)** - Density and Grid based
- **PROCLUS (SIGMOD 99)** - modification to k-means algorithm

Require key user i/p parameters - hard to decide!
Subspace Clustering

- **Observation:** "If a collection of points S is a cluster in a k-dimensional space, then S is also a part of a cluster in any (k-1) dimensional projection of the space”

- **Algorithm:** candidate dense units (CDU) in any k dimensions are obtained by merging dense units in (k-1) dimensions which share first (k-2) dimensions. *(CLIQUE)*
  - Populate CDUs by a pass on data set.
  - Find dense units in each dimension (use thresholds).
  - Combine dense units to form higher dimension CDUs.
  - Terminates when no more CDUs.
Adaptive Grids

- Automatic Grid fitting based on data distribution
  - MAFIA: Merging of Adaptive Finite Intervals!
- Optimal Bins in each dimension - very few CDUs
  (a): CLIQUE
  (b): MAFIA
Base MAFIA Algorithm

- Divide each dimension into fine regions.
- Compute histogram in these regions in each dimension.
- A sliding window to find max histogram value in window.
- Adjacent units with nearly same histogram values merged together to form larger bins.
- Threshold of each bin formed computed automatically.

A bin with histogram value much greater (by a factor 2~3) than equi-distribution of data is DENSE.
MAFIA Algorithm (merging dense units)

Algorithm: Candidate dense units in any k dimensions are obtained by merging dense units in \((k-1)\) dimensions which share any \((k-2)\) dimensions.

Ex: \((\{a1,c7,b8\}, \{c7,b8,d9\}) \rightarrow \{a1,c7,b8,d9\}\)
• **CLIQUE**: CDUs in any dimension $k$ is formed by combining dense units of dimension $(k-1)$ which share *first* $(k-2)$ dimensions.

• **pMAFIA**: CDUs in any dimension $k$ is formed by combining dense units of dimension $(k-1)$ which share *any* $(k-2)$ dimensions.

Data dimension $(d) = 10$, current dimension $(k) = 5$, $P(e) = 93.3\%$

$$\text{Probability}\{\text{error}\} = 1 - \frac{\sum_{i=2}^{n-2} \sum_{i=2}^{n-3} \sum_{i=2}^{n-2} \sum_{i=2}^{n-3} \frac{i(i^2 - 1)}{6} \begin{pmatrix} d \choose k-1 \end{pmatrix} \begin{pmatrix} d-k+1 \choose 2 \end{pmatrix}}{\begin{pmatrix} d \choose k-1 \end{pmatrix} \begin{pmatrix} d-k+1 \choose 2 \end{pmatrix}}$$
Parallel MAFIA

- Completely *Unsupervised Data Mining Algorithm.*
  - Requires no user input!
- Scalable in data size and number of dimensions
- Parallelization provides speedup for the subspace clustering algorithm.

choudhary@ece.nwu.edu

Alok Choudhary
**pMAFIA Algorithm:**

Each proc reads in *data parallel fashion* // in chunks: *out-of-core*
Construct histogram in each dimension.
Reduce communication, obtain global histogram.
All processors build *Adaptive grids* using histogram.
Current Dimension $k = 1$;
while (no more dense units found)
  if ($k > 1$)
    **Build-Candidate-Dense-units();** //task parallel
    Populate the CDUs in *data parallel* with chunking of data set
Reduce communication, obtain global CDU population.
**Identify-dense-units();** //task parallel
**Build-dense-unit-data-structures();** //task parallel
Build-Candidate-Dense-Units

Current Dimension(k) = 3.

Dense Unit Dimensions

Dense Unit Bins

CDU Dimensions

CDU Bins

Ndu = 8

Repeat

Ncdu = 7

choudhar@ece.nwu.edu

Alok Choudhary
Task Parallelism

- **Identify-Dense-Units()**
  - A CDU is dense if all bins forming the CDU are dense.
  - large Ncdu: each proc processes Ncdu/p CDUs.

- **Build-Dense-Unit-Data-Structures()**
  - large Ndu: data structures constructed in parallel.
    - A set of dimensions and corresponding bin indices in those dimensions characterize a Dense Unit.
**Build-Candidate-Dense-Units**

- \( O(Ndu^2) \) algorithm: compare each DU with every other DU.

\[
Ndu + (n_{i+1} - n_i) - (\sum_{j=a_i}^{a_i+1} j) = \frac{Ndu(Ndu + 1)}{2p}
\]

- \( O(Ncdu^2) \) algorithm: eliminate repeat CDUs.
Optimizations:
- ‘d’ dimensional DU requires just 2d bytes of memory.
- **Data structures**: linear array of bytes $\Rightarrow$ small message buffers & space optimized.

$$O(c^k + (N/pB) \times k \times Tio + T_{comm} \times S \times p \times k)$$
(a) **CLIQUE**:

- Loss of quality: reports ‘pqr’s’ as the cluster!
- Requires a complicated post processing step.
- Bin selection and threshold fixing is a non trivial problem. Cannot validate results.

(b) **MAFIA**: *Almost exact cluster boundaries recognized*

- No post processing step required.
<table>
<thead>
<tr>
<th></th>
<th>Cluster Dimensions</th>
<th>Clusters Discovered</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CLIQUE (fixed 10 bins)</strong></td>
<td>{1, 7, 8, 9}, {2, 3, 4, 5}</td>
<td>{1, 7, 8, 9}, {2, 3, 4, 5}</td>
</tr>
<tr>
<td><strong>CLIQUE (variable bins)</strong></td>
<td>{1, 7, 8, 9}, {2, 3, 4, 5}</td>
<td>{2, 3, 4, 5}</td>
</tr>
<tr>
<td><strong>pMAFIA</strong></td>
<td>{1, 7, 8, 9}, {2, 3, 4, 5}</td>
<td>{1, 7, 8, 9}, {2, 3, 4, 5}</td>
</tr>
</tbody>
</table>

400,000 records in 10 dimensions; 2 clusters in 2 4d subspaces.
Adaptive Grids

300,000 records in a 15 Dimension space with 1 cluster of 5 dimensions.

- A speedup of 80 obtained over CLIQUE.
- CLIQUE failed to produce results with our modified CDU generation algorithm even in 2 hours on 16 processors.
- This relatively small data set mined in just 32 seconds on 1 processor.
Scalability with Data Set Size

- 20 Dimension data with 5 clusters in 5 different subspaces
- up to 11.8 million records
- Clusters detected in just about 3 minutes on 16 processors!
- Almost Linear with the increase in the data set size
Parallelization (on IBM SP2)

- 30 Dimension data with 8.3 million records, 5 clusters each in a 6 dimension subspace.
- Near linear speedups
- Negligible Communication overheads (<1%)

choudhar@ece.nwu.edu

Alok Choudhary
Data Dimensionality

- 250,000 records, 3 clusters in different 5 dimensional subspaces.
- Near linear behavior with data dimensionality: Algorithm depends on the maximum number of dimensions in a clusters and not the data dimensionality.
Cluster Dimensionality

- 50 dimension data with 1 cluster, 650,000 records; cluster dimension from 3 to 10.
- Behavior in line with the order of the algorithm: increases with subspace coverage of the cluster.
Scalability on Movie Data

72,916 users rated 1628 movies in 18 months: **2.8 Million ratings**

4D data: \{user-id, movie-id, weight, score\}

<table>
<thead>
<tr>
<th>Processors</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run Times (in sec)</td>
<td>144.86</td>
<td>70.47</td>
<td>36.86</td>
<td>20.35</td>
<td>10.18</td>
</tr>
<tr>
<td>Speed Up</td>
<td>1</td>
<td>2.06</td>
<td>3.93</td>
<td>7.11</td>
<td>14.23</td>
</tr>
</tbody>
</table>

**Discovered seven interesting 2d clusters!**
German Stock Exchange Data

- DAX prediction data set was based on a 12 input time series like stock indices, bond indices, gold prices, etc.
- 22 dimensions with 2757 records: Major gains from task parallelism.
- Mined clusters in 8.16 seconds on 8 processors.
- Unique clusters discovered in 3, 4, 5 and 6 dimensional subspaces.
Conclusions

- **pMAFIA**: Completely unsupervised subspace clustering algorithm
  - *Adaptive grids*: vast improvement both in computation time & cluster quality.
  - *No user input required*. Grid size and thresholds automatically determined.
  - *Modification of the CDU generation procedure*.

- **First parallel subspace clustering algorithm**
  - Both task and data parallelism.
  - Scalable to massive data sets and data dimensions.
  - Near linear speedups achieved.
  - Negligible Communication Overheads.