Advanced Algorithmics (6EAP)
Clustering and Seriation

Jaak Vilo
2012 Spring

Vehicle Example

<table>
<thead>
<tr>
<th>Vehicle</th>
<th>Top speed [km/h]</th>
<th>Colour</th>
<th>Air resistance</th>
<th>Weight [Kg]</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>220</td>
<td>red</td>
<td>0.30</td>
<td>1400</td>
</tr>
<tr>
<td>V2</td>
<td>230</td>
<td>black</td>
<td>0.32</td>
<td>1500</td>
</tr>
<tr>
<td>V3</td>
<td>260</td>
<td>red</td>
<td>0.29</td>
<td>1500</td>
</tr>
<tr>
<td>V4</td>
<td>140</td>
<td>gray</td>
<td>0.35</td>
<td>800</td>
</tr>
<tr>
<td>V5</td>
<td>135</td>
<td>blue</td>
<td>0.33</td>
<td>950</td>
</tr>
<tr>
<td>V6</td>
<td>130</td>
<td>white</td>
<td>0.40</td>
<td>600</td>
</tr>
<tr>
<td>V7</td>
<td>100</td>
<td>black</td>
<td>0.50</td>
<td>3000</td>
</tr>
<tr>
<td>V8</td>
<td>105</td>
<td>red</td>
<td>0.60</td>
<td>2500</td>
</tr>
<tr>
<td>V9</td>
<td>110</td>
<td>gray</td>
<td>0.55</td>
<td>3500</td>
</tr>
</tbody>
</table>

Vehicle Clusters

Terminology

Object or data point
Feature
Cluster
Feature space
Label

Machine learning

Typical approaches

- Clustering ("Unsupervised learning")
Typical approaches

- Regression, classification ("Supervised learning")

Unsupervised learning vs. Supervised

- Find groups inherent to data (clustering)
- Find a “classifier” for known classes

Fuzzy vs Hard

- Each object belongs to each cluster with some weight (the weight can be zero)
- Each object belongs to exactly one cluster

Typical approaches

- Outlier detection

Motivation: Why Clustering?

Problem: Identify (a small number of) groups of similar objects in a given (large) set of object.

Goals:
- Find representatives for homogeneous groups → Data Compression
- Find “natural” clusters and describe their properties → “natural” Data Types
- Find suitable and useful grouping → “useful” Data Classes
- Find unusual data object → Outlier Detection
Clustering

- An old problem
- Many methods
- No single best “suites all needs” method

---

Clustering – it’s “easy” (for humans)

- Edge-Detection (advantage to smooth contours)
- Texture clustering

---

Clustering cont...
Group objects by **similarity**

- What means similarity?
  - Similar features

- Distance
  - Small distance ~ high similarity
  - Low similarity ~ large distance

Distance measures:
which two profiles are similar to each other?

<table>
<thead>
<tr>
<th>Method</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean</td>
<td>( d(f, g) = \sqrt{\sum_{i=1}^{c} (f_i - g_i)^2} )</td>
</tr>
<tr>
<td>Euclidean squared</td>
<td>( d(f, g) = \sum_{i=1}^{c} (f_i - g_i)^2 )</td>
</tr>
<tr>
<td>Manhattan (city-block)</td>
<td>( d(f, g) = \sum_{i=1}^{c}</td>
</tr>
<tr>
<td>Average distance</td>
<td>( d(f, g) = \frac{1}{c} \sum_{i=1}^{c} (f_i - g_i)^2 )</td>
</tr>
</tbody>
</table>

Some standard distance measures

Pearson correlation

\[
d(f, g) = 1 - \frac{\sum_{i=1}^{c} (f_i - \bar{f})(g_i - \bar{g})}{\sqrt{\sum_{i=1}^{c} (f_i - \bar{f})^2 \sum_{i=1}^{c} (g_i - \bar{g})^2}}
\]

If means of each column are 0, then it becomes:

\[
d(f, g) = 1 - \frac{\sum_{i=1}^{c} f_i g_i}{\sqrt{\sum_{i=1}^{c} f_i^2 \sum_{i=1}^{c} g_i^2}} = 1 - \cos \Theta
\]

Chord distance

\[
d(f, g) = \sqrt{2(1 - \cos \Theta)}
\]

Euclidean distance between two vectors whose length has been normalized to 1.
9.5.2012

Rank correlation

\[ d(f,g) = 1 - \frac{6 \sum_{i=1}^{c} (\text{rank}_f - \text{rank}_g)}{c(c-1)} \]

- Rank - smallest has rank 1, next 2, etc.
- Equal values have rank that is average of the ranks

<table>
<thead>
<tr>
<th>( f )</th>
<th>3</th>
<th>17</th>
<th>12</th>
<th>12</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{rank} )</td>
<td>1</td>
<td>5</td>
<td>3.5</td>
<td>3.5</td>
<td>2</td>
</tr>
</tbody>
</table>

Clustering algorithms

- Similar objects should belong to same cluster

Hierarchical clustering

- Assign each object into a singleton cluster
- Calculate all pairwise distances
- While more than 1 cluster
  - select a pair with smallest distance
  - merge the two clusters
  - update the changed distances after merger

Update distances

- Merge \( \text{Ca}, \text{Cb} \) into \( C \)
- Re-calculate all distances \( D(C_i, C) \)

Hierarchical clustering

Performance: \( O(dn^2) \)

1. All against all distance matrix
2. Linkage strategy - identify closest clusters and merge

Cluster matrices:

Cluster sequences:
Merge Ca, Cb into C

• $D(C_i, C) = \min\{ D(C_i, Ca) , D(C_i, Cb) \}$
  - Single link; Minimal distance

• $D(C_i, C) = \max\{ D(C_i, Ca) , D(C_i, Cb) \}$
  - Complete link; Maximum distance

• $D(C_i, C) = \text{average}\{ D(C_i, Ca) , D(C_i, Cb) \}$
  - $n_i/(n_a+n_b) * D(C_i, Ca) + n_b/(n_a+n_b) * D(C_i, Cb)$
  - UPGMA – Unweighted Pair Group Method Average

How to Define Inter-Cluster Similarity

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward’s Method uses squared error

UPGMA

\[
D(C_k, (C_i \cup C_j)) = \frac{|C_i|}{|C_i| + |C_j|} D(C_k, C_i) + \frac{|C_j|}{|C_i| + |C_j|} D(C_k, C_j)
\]

WPGMA:

\[
D(C_k, (C_i \cup C_j)) = \frac{1}{2} (D(C_k, C_i) + D(C_k, C_j))
\]

Single-Linkage:

\[
D(C_k, (C_i \cup C_j)) \leftarrow \min \{ D(C_k, C_i), D(C_k, C_j) \}
\]
How to Define Inter-Cluster Similarity

- **MIN**
- **MAX**

Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function
  - Ward’s Method uses squared error

Proximity Matrix

- **p₁**
- **p₂**
- **p₃**
- **p₄**
- **p₅**

Input data visualized

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(3.5, 0.9)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(1.5, 3.6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(1.5, 5.8)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(2.1, 4.6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(3.4, 1.2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(5.1, 5.2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(4.4, 5.5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(4.3, 2.2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(8.3, 1.4)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Euclidean distances

Average linkage hierarchical clustering

Input data

Heatmap

- **TAGCCCA**
- **TAGCCCA**
- **TAGCCCA**
- **TAGCCCA**
- **TAGCCCA**
- **TAGCCCA**

UT: Data Mining 2009
Running Jme for hierarchical clustering

Clustering

10,100, 1000 dim

Distances 10 attrib.

Distances 100 attrib.

Time in seconds

1 minute

10K

20K

Data size

Examples of Hierarchical Clustering in Bioinformatics

Time

• $O(n^2)$ distances

• n-1 times merge
  – select smallest distance
  – update all distances to new cluster

Examples of Hierarchical Clustering in Bioinformatics

Gene expression clustering

Phylogeny
9.5.2012

Design any heat-map coloring scheme

Heat map color schema design

6200 genes, 80 exp.

Monitor size 1600x1200 pixels

Laptop: 800x600

VisHIC, 2009

6200 genes, 80 exp.

Monitor size 1600x1200

Laptop: 800x600

“COLLAPSE”

75 subtrees

Developed and implemented in Expression Profiler in October 2000 by

Jaak Vilo
Fast Approximate Hierarchical Clustering using Similarity Heuristics

Hierarchical clustering is applied in gene expression data analysis, number of genes can be 20000+

Hierarchical clustering:

Each subtree is a cluster.

Hierarchy is built by iteratively joining two most similar clusters into a larger one.

Fast Hierarchical Clustering

Avoid calculating all \(O(n^2)\) distances:

- Estimate distances
- Use pivots
- Find close objects
- Cluster with partial information


Input data

<table>
<thead>
<tr>
<th>i</th>
<th>(x_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0.5, 0.9)</td>
</tr>
<tr>
<td>2</td>
<td>(1.5, 3.0)</td>
</tr>
<tr>
<td>3</td>
<td>(1.7, 5.2)</td>
</tr>
<tr>
<td>4</td>
<td>(2.1, 8.6)</td>
</tr>
<tr>
<td>5</td>
<td>(3.0, 8.4)</td>
</tr>
<tr>
<td>6</td>
<td>(3.1, 1.2)</td>
</tr>
<tr>
<td>7</td>
<td>(8.4, 6.8)</td>
</tr>
<tr>
<td>8</td>
<td>(8.5, 2.2)</td>
</tr>
<tr>
<td>9</td>
<td>(8.3, 1.4)</td>
</tr>
</tbody>
</table>

Input data visualized

Euclidean distances

<table>
<thead>
<tr>
<th>(d_{x_i, x_j})</th>
<th>0.0</th>
<th>5.0</th>
<th>5.0</th>
<th>5.0</th>
<th>10.0</th>
<th>15.4</th>
<th>15.4</th>
<th>15.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d_{x_i, x_j})</td>
<td>5.0</td>
<td>0.0</td>
<td>5.0</td>
<td>5.0</td>
<td>10.0</td>
<td>15.4</td>
<td>15.4</td>
<td>15.4</td>
</tr>
</tbody>
</table>
| Average linkage hierarchical clustering

Distances from one pivot

Distances from two pivots

Distances from two pivots

Here we use Chebyshev distance (maximum of differences)

By triangle inequality we get:
Euclidean distance in original plot cannot be smaller than Chebyshev distance here
9.5.2012

Major Clustering Approaches

- **Partitioning algorithms**/Representative-based/Prototype-based Clustering Algorithm: Construct various partitions and then evaluate them by some criterion or fitness function \( \rightarrow \text{k-means} \)
- **Hierarchical algorithms**: Create a hierarchical decomposition of the set of data (or objects) using some criterion
- **Density-based**: based on connectivity and density functions \( \rightarrow \text{DBSCAN}, \text{DENCLUE}, \ldots \)
- **Grid-based**: based on a multiple-level granularity structure
- **Model-based**: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other \( \rightarrow \text{EM} \)

**Representative-Based Clustering**

- Aims at finding a set of objects among all objects (called representatives) in the data set that best represent the objects in the data set. Each representative corresponds to a cluster.
- The remaining objects in the data set are then clustered around these representatives by assigning objects to the cluster of the closest representative.

Remarks:

1. The popular k-medoid algorithm, also called PAM, is a representative-based clustering algorithm; k-means also shares the characteristics of representative-based clustering, except that the representatives used by k-means not necessarily have to belong to the dataset.
2. If the representative do not need to belong to the dataset we call the algorithms prototype-based clustering. K-means is a prototype-based clustering algorithm

**K-means, K-medoids, ...**

- Partition the data points into K groups
  - Each group is centered around it’s mean or medoid
- Re-calculate the “center” and repeat

- Mean is an “abstract point”
- Medoid: most central object

**The K-Means Clustering Method**

- Example

1) Datapoints sorted according to EGO and
2) Each point is compared with the later points until one hypercube away

Epsilon Grid Order (EGO)
The K-Means Clustering Method

- Given $k$, the $k$-means algorithm is implemented in 4 steps:
  1. Partition objects into $k$ nonempty subsets
  2. Compute seed points as the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.
  3. Assign each object to the cluster with the nearest seed point.
  4. Go back to Step 2, stop when no more new assignment.

K-means

1. Guess $k$ centres
2. Assign obj to clusters
3. Move C to gravity centres

Comments on K-Means

**Strength**
- Relatively efficient: $O(kdn^t)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations, $d$ is the # dimensions. Usually, $d, k, t << n$; in this case, K-Mean’s runtime is $O(n)$.
- Storage only $O(n)$—in contrast to other representative-based algorithms, only computes distances between centroids and objects in the dataset, and not between objects in the dataset; therefore, the distance matrix does not need to be stored.
- Easy to use; well studied; we know what to expect.
- Finds local optimum of the SSE fitness function. The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms.
- Implicitly uses a fitness function (finds a local minimum for SSE see later) — does not waste time computing fitness values.

**Weakness**
- Applicable only when mean is defined — what about categorical data?
- Need to specify $k$, the number of clusters, in advance.
- Sensitive to outliers.
- Not suitable to discover clusters with non-convex shapes.
- Sensitive to initialization; bad initialization might lead to bad results.

Objective of RSC: Find a subset $O_R$ of $O$ such that the clustering $X$ obtained by using the objects in $O_R$ as representatives minimizes $q(X)$; $q$ is an objective/fitness function.
Complication: Empty Clusters

We assume that the k-means initialization assigns the green, blue, and brown points to a single cluster; after centroids are computed and objects are reassigned, it can easily be seen that the brown cluster becomes empty.

Convex Shape Cluster

- Convex Shape: if we take two points belonging to a cluster then all the points on a direct line connecting these two points must also in the cluster.
- Shape of K-means/K-mediods clusters are convex polygons Convex Shape.
- Shapes of clusters of a representative-based clustering algorithm can be computed as a Voronoi diagram for the set of cluster representatives.
- Voronoi cells are always convex, but there are convex shapes that a different from those of Voronoi cells.

Voronoi Diagram for a Representative-based Clustering

Each cell contains one representatives, and every location within the cell is closer to that sample than to any other sample. A Voronoi diagram divides the space into such cells. Voronoi cells define cluster boundary!

K-means

- Finds local optimum
  - vary many times with random start
  - make an “educated guess” to start with
    - e.g. sample the data, perform hierarchical clustering, select K “centers”.

K-medoids

• Choose the cluster center to be one of the existing objects.
• **Why?**
  • If more complex data or distance measure the “Real” center could not be found easily
  • What is the mean of categorical data?
    – yellow, red, pink?
  • Instead of trying to “invent” – use one of the existing objects, whatever the distance measure

**A Gaussian Mixture Model for Clustering**
- Assume that data are generated from a mixture of Gaussian distributions
- For each Gaussian distribution
  - Center: \( \mu \)
  - Variance: \( \Sigma \)
  - Probability: \( x \)
- \( N \) data points

**Clustering – 2 dimensions**

**Bi-clustering**

• Traditional clustering will be driven by “global” signals and ignore “local” signals
• Bi-clustering identifies groups of genes and conditions rather than just genes

**Bi-clustering goals/issues**

• Better capture biological reality
  – Genes only cooperate in certain conditions
  – Genes can have multiple functions
  – Datasets have functional biases
• Computationally difficult problem
  – Reducible to bi-clique finding
    • NP-complete
• Heuristics, simplifications, approximations
  – e.g. \( \delta \)-biclusters, SAMBA, PISA

**Self Organising Maps (SOM)**

MaN matrix of neurons, each representing “a cluster”
Object \( X \) is put to \( W \), to which it is most similar
\( W \) and its near surrounding is changed to resemble \( X \) more
Train, train, train...
Motivation: The Problem Statement

- The problem is how to find out semantics relationship among lots of information without manual labor
  - How do I know, where to put my new data in, if I know nothing about information's topology?
  - When I have a topic, how can I get all the information about it, if I don’t know the place to search them?

Example: Self-Organizing Maps
Animal names and their attributes

Self-Organizing Maps: Origins

Self-Organizing Maps
- Ideas first introduced by C. von der Malsburg (1973), developed and refined by T. Kohonen (1982)
- Neural network algorithm using unsupervised competitive learning
- Primarily used for organization and visualization of complex data
- Biological basis: 'brain maps'

Self-Organizing Maps

SOM – Architecture
- Lattice of neurons (nodes) accepts and responds to set of input signals
- Responses compared; 'winning' neuron selected from lattice
- Selected neuron activated together with 'neighbourhood' neurons
- Adaptive process changes weights to more closely resemble inputs

SOM – Result Example

Helsinki University of Technology
9.5.2012

Motivation: The Idea

- Text objects must be automatically produced with semantics relationships

Web Document Clustering Using SOM

- The result of SOM clustering of 12088 Web articles
- The picture on the right: drilling down on the keyword "mining"
- Based on websom.hut.fi

SOM - algorithm

- Iterate through data points (examples)
  - Find closest point in SOM grid
  - Adjust the representative vector and local neighbourhood

Finding a Winner

(iii) Find the best-matching neuron ω(x), usually the neuron whose weight vector has smallest Euclidean distance from the input vector x

The winning node is that which is in some sense 'closest' to the input vector 'Euclidean distance' is the straight line distance between the data points, if they were plotted on a (multi-dimensional) graph

Euclidean distance between two vectors a and b, a = (a₁, a₂, …, aₙ), b = (b₁, b₂, …, bₙ), is calculated as:

\[ d_{ab} = \sqrt{\sum (a_i - b_i)^2} \]

Clustering etc. algorithms

- Hierarchical clustering methods + visualisation
- K-means, Self Organising Maps (SOM)
- SOTA trees (Self Organising Maps + Tree)
- Fuzzy, EM (object can belong to several clusters)
- Graph theory (cliques, strongly connected components)
- Similarity search: X -> Y s.t. d(X, Y) < 0.3
- Model based (rediscover distributions)
- Planar embeddings, Multidimensional scaling
- Principal Component Analysis
- Correspondence analysis
- Independent Component Analysis
Similarity searches

Query: “cyc1” (cyc1, activator for cyc1, repressor for cyc1)
=> 3 genes + 10 most similar ones for each
= 3 “clusters”

EM — Expectation Maximization

- EM — A popular iterative refinement algorithm
- An extension to k-means
  - Assign each object to a cluster according to a weight (prob. distribution)
  - New means/covariances are computed based on weighted measures
- General idea
  - Starts with an initial estimate of the parameter vector
  - Iteratively rescores the patterns against the mixture density produced by the parameter vector
  - The rescored patterns are used to update the parameter updates
  - Patterns belonging to the same cluster, if they are placed by their scores in a particular component
- Algorithm converges fast but may not be in global optima

The EM (Expectation Maximization) Algorithm

- Initially, randomly assign k cluster centers
- Iteratively refine the clusters based on two steps
  - Expectation step: assign each data point $X_i$ to cluster $C_k$ with the following
    \[ P(X_i \in C_k) = \frac{p(C_k)p(X_i|C_k)}{p(X_i)} \]
  - Maximization step:
    - Estimation of model parameters
      \[ m_k = \frac{1}{N} \sum_{i=1}^{N} X_i | X_i \in C_k \]
- PCA (Principal Component Analysis)
  - Also called SVD (Singular Value Decomposition)
- Reduces dimensionality of gene expression space
- Finds best view that helps separate data into groups

Other Clustering Methods

- Supervised Methods
  - SVM (Support Vector Machine)
  - Previous knowledge of which genes expected to cluster is used for training
  - Binary classifier uses ‘feature space’ and ‘kernel function’ to define an optimal hyperplane
  - Also used for classification of samples – ‘expression fingerprinting’ for disease classification

Persistent Systems Pvt. Ltd.
http://www.persistent.co.in
The Curse of Dimensionality
(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed
- Adding a dimension “stretch” the points across that dimension, making them further apart
- Adding more dimensions will make the points further apart—high dimensional data is extremely sparse
- Distance measure becomes meaningless—due to equi-distance

Why Subspace Clustering?
(adapted from Parsons et al. SIGKDD Explorations 2004)

- Clusters may exist only in some subspaces
- Subspace-clustering: find clusters in all the subspaces

DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point.
Density – no particular “shape”

DBSCAN: Core, Border, and Noise Points

DBSCAN Algorithm

- Eliminate noise points
- Perform clustering on the remaining points

```
current_cluster_label = 1
for all core points do
    if the core point has no cluster label then
        current_cluster_label = current_cluster_label + 1
        label the current core point with cluster label current_cluster_label
    end if
    for all points in the Eps-neighborhood, except the point itself do
        if the point does not have a cluster label then
            label the point with cluster label current_cluster_label
        end if
    end for
end for
```

When DBSCAN Works Well

- Resistant to Noise
- Can handle clusters of different shapes and sizes

When DBSCAN Does NOT Work Well

- Varying densities
- High-dimensional data
DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their kth nearest neighbors are at roughly the same distance
- Noise points have the kth nearest neighbor at farther distance
- So, plot sorted distance of every point to its kth nearest neighbor

Summary

- Cluster analysis groups objects based on their similarity and has wide applications
- Measure of similarity can be computed for various types of data
- Clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- Outlier detection and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches
- There are still lots of research issues on cluster analysis
Summary: Goals of clustering

- Scalability
- Deal with different types of attributes
- Clusters of arbitrary shape
- Minimal expert knowledge on parameters
- Deal with noise
- Incremental and not dependent on order
- High dimensionality
- Constraints
- Interpretability and usability

Problems and Challenges

- Considerable progress has been made in scalable clustering methods
  - Partitioning: k-means, k-medoids, CLARANS
  - Hierarchical: BIRCH, ROCK, CHAMELEON
  - Density-based: DBSCAN, OPTICS, DenClue
  - Grid-based: STING, WaveCluster, CLIQUE
  - Model-based: EM, Cobweb, SOM
  - Frequent pattern-based: pCluster
  - Constraint-based: COD, constrained-clustering
- Current clustering techniques do not address all the requirements adequately, still an active area of research

Seriation, Clustering and Matrix Reordering: Towards the Encyclopedia of Structures

INNAR LIIV
innar.liiv@ttu.ee

Visualization and human computation “brain exercise”

Perception of/and experience

- Simple example of 5 entities (persons) and their relationships
- Who would you prefer to be?
- Who wouldn’t you want to be?
- And what if the relationship means “company A sells to company B”?
- What if relationship means “love”?

Outline

- Micro-tutorial on Seriation:
  - What is it? Who cares?
  - Clustering versus seriation;
  - Related work and background
  - Recent advances
- Similarity (“goodness”) measures
- Where to go from here?
  - The Encyclopedia (Gallery, DB) of Structures?
9.5.2012

Definition: Seriation

- Seriation is an exploratory combinatorial data analysis technique to reorder objects into a sequence along a one-dimensional continuum so that it best reveals regularity and patterning among the whole series.

A simple example with 11 objects

“Raw data”

Seriation vs clustering

Seriation and Matrix Reordering

- Seriation is typically applied for matrix reordering (two-way one/two-mode seriation);
- Every matrix is two-way*, N x N matrix is one-mode and N x M matrix is two-mode.

*Using Carroll-Arabie taxonomy of scaling methods and terminology of Tucker
Seriation vs Clustering

- Example by Prof. Gilles Caraux (Permutmatrix software):

  ![Original Data Matrix](image1)
  ![After Cluster Analysis of Rows](image2)
  ![After Seriation](image3)

Can you see the pattern in data? Did you see it?

Seriation: who cares (nowadays)?

- Information visualization & HCI community;
  - Ghoniem&Fekete&Castagliola: A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations + MatrixExplorer (Henry, Fekete)

- Data mining and statistics community;
- Bioinformatics community;
- Social Network Analysis community;
- Operations research and combinatorial optimization community.

Related work

- Robinson Archaeology
- Sokal Biology
- Burbidge Manufacturing
- Hartigan Statistics
- Mullad Võhandu Survey DA
- Marcotorchino Unified approach
- Siirtola Visualization
- Chen et al. Data mining
- Liiv Unified view Algorithms Applications
- Forsyth&Sokol Sociology
- Berin Cartography
- McCormick Op.research
- Czekanowski Anthropology
- 1991
- 1992
- 1994
- 2004
Recent advances

- Niermann (2005) presented a GA approach for seriation in *The American Statistician*

“Optimal leaf ordering (OLO)”

- Bar-Joseph et al. 2001:

> Unclassed Matrix Shading and Optimal Ordering in Hierarchical Cluster Analysis (Gale et al., *Journal of Classification*, 1:75-92, 1984)

No name consensus

- Czekanowski diagram, Robinson matrix, Reorderable matrix, Matrix reordering, Matrix visualization, Matrix analysis, Matrix permutation, Permutable matrix, Array-based clustering, Block clustering, Biclustering (two(n)-mode clustering), Co-clustering, Product Flow Analysis, Group Technology, Part/Machine group formation, Manufacturing cell formation, Cellular manufacturing, Seriation, cleaned up differential shading of the similarity matrix, Matrix tile analysis, Rearrangement clustering, Generalized Association Plots (GAP), non-destructive data analysis, optimal order of matrices, Optimal leaf ordering, band form, banded structure; Matrice ordonnable; Differentialdiagnose; метод групповой технологии и организации группового производства;

Seriation: a unified view

- unidimensional seriation
- block diagonal seriation
- block checkboard seriation
- Pareto seriation

Similarity (“goodness”) measures

- McCormick et al. (1969, 1972):
  \[ \text{argmin}_{\Phi} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \Phi_{ij} (\Phi_{i,i+1} + \Phi_{i+1,i} + \Phi_{i,i-1} + \Phi_{i+1,i+1}) \]
  - Cumulative Hamming (Verin/Grishin,1986):
  \[ \text{argmin}_{\Phi} \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} d(x_F(i), x_F(i+1)) \right)^2 \times \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} d(x_F(i), x_F(i+1)) \right)^{-1} \]
  - Can be generalized and written compactly:
  \[ \text{argmin}_{\Phi} L(\Pi\Phi) \]

4 Important questions

- I’m not buying that matrix representation is better than graph layout! (actually, it’s worse!!!)
- How is it different from correlation? (why can’t I just calculate corr coef for everything and sort as a list?)
- How is it different from clustering? (there’s lots of tools for clustering – Why can’t I just pick one of those?)
- What is the added value to InfoVis community from this approach?
I’m not buying that matrix representation is better than graph layout!

- Such discussion is older than us (Forsyth-Katz vs Moreno 1940s, recent user studies by Fekete, Henry, Ghoniem)
- “Clique”, clusters, hubs, chains harder to detect in graphs with different entity types (“bipartite” and n-partite graphs)
- Hard to read if n gets bigger
- With graphs we encode only positive connections (existing relationships)

How is it different from correlation? (why can’t I just calculate corr coef for everything and sort as a list?)

- We don’t know the two attributes!
- We want to find multiple correlations (corr between >> more than two attr.) (2n list)
- We don’t always know what “level” of correlation provides the most information (highest corr != “best” corr)
- We are interested in chained corr
- Not to mention that there are some fundamental issues already with std corr:

How is it different from clustering? (there’s lots of tools for clustering – Why can’t I just pick one of those?)

- Algorithmic problem: k # of clusters unknown
- Goal of clustering is to assign similar entities to groups, not to identify or describe similarities/affinities between entities!
- Clustering of attributes aka/~ factor analysis
- It is not a clustering’s “fault”, because if the goal is not to find all similarities between entities and between clusters, it would be inefficient extra work for CPU/GPU
What is the added value to InfoVis community from this approach?

- Important to distinguish that learning to see different structural patterns from the overview is not just moving along the learning curve to get the technique, but to accumulate knowledge from all your previous works.
- Not just learning how to read the display, but how to connect and combine with past experience, background information, memories from previous investigations, not on entity level, but @ abstract metalevel.

EXAMPLE: QUERY2

School violence

1) Bottleneck machine (in manufacturing)
2) Excellent position (in supply chain)
3) Miserable love (Psychology)

THANK YOU