VK Multimedia Information Systems

Mathias Lux, mlux@itec.uni-klu.ac.at

Dienstags, 16.3o Uhr

Agenda

- Evaluations
- Local features
- Bag of visual words
- Clustering

Evaluations

- Wang SIMPLIcity data set
  - 10 categories á 100 images - 1,000 images in total
  - Photographs from the Corel Stock Photos
- Uncompressed Colour Image Database (UCID)
  - 1,338 images and ~260 queries
  - Photographs

Evaluations

- LIRE on the SIMPLIcity data set

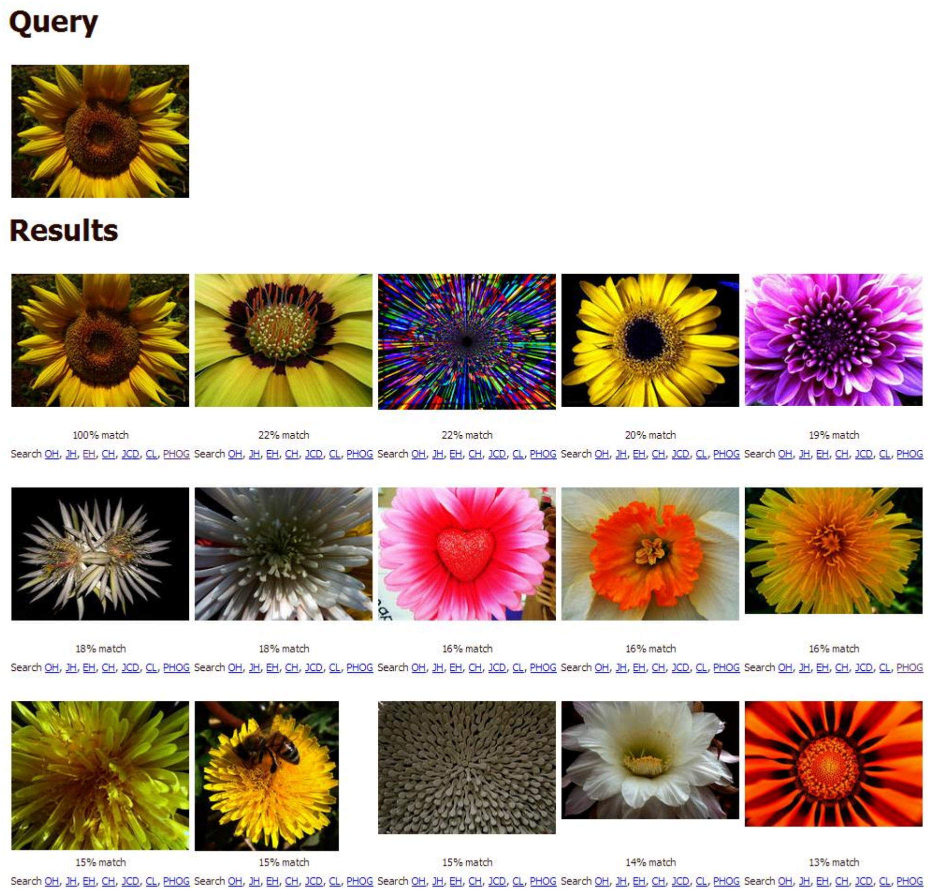
Feature	map	p@10	er
CEDD	0.507	0.710	0.195
Color Correlogram	0.498	**0.740**	**0.163**
Color Layout	0.439	0.612	0.307
Edge Histogram	0.333	0.500	0.403
FCTH	0.501	0.701	0.202
JCD	**0.514**	0.722	0.184
Joint Histogram	0.449	0.689	0.201
Opponent Histogram	0.450	0.635	0.274
PHOG	0.352	0.535	0.370
RGB Color Histogram	0.450	0.705	0.194
Scalable Color	0.305	0.470	0.464

Evaluations

- LIRE on the UCID data set

Feature	map	p@10	er
CEDD	0.442	0.427	0.531
Color Correlogram	**0.585**	**0.480**	**0.370**
Color Histogram	0.403	0.356	0.550
Color Layout	0.277	0.285	0.679
Edge Histogram	0.180	0.202	0.813
FCTH	0.452	0.416	0.527
JCD	0.466	0.430	0.515
Joint Histogram	0.348	0.313	0.603
Opponent Histogram	0.319	0.308	0.649
PHOG	0.238	0.238	0.714
Scalable Color	0.175	0.184	0.836

Evaluations
PHOG use case on MIRFLICKR



Evaluations

Edge Histogram use case on logo data

Query

Results

100% match	21% match	19% match	18% match	16% match
Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG
15% match	13% match	13% match	13% match	12% match
Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG	Search OH, JH, EH, CH, JCD, CL, PHOG

Agenda

- Evaluations
- **Local features**
- Bag of visual words
- Clustering

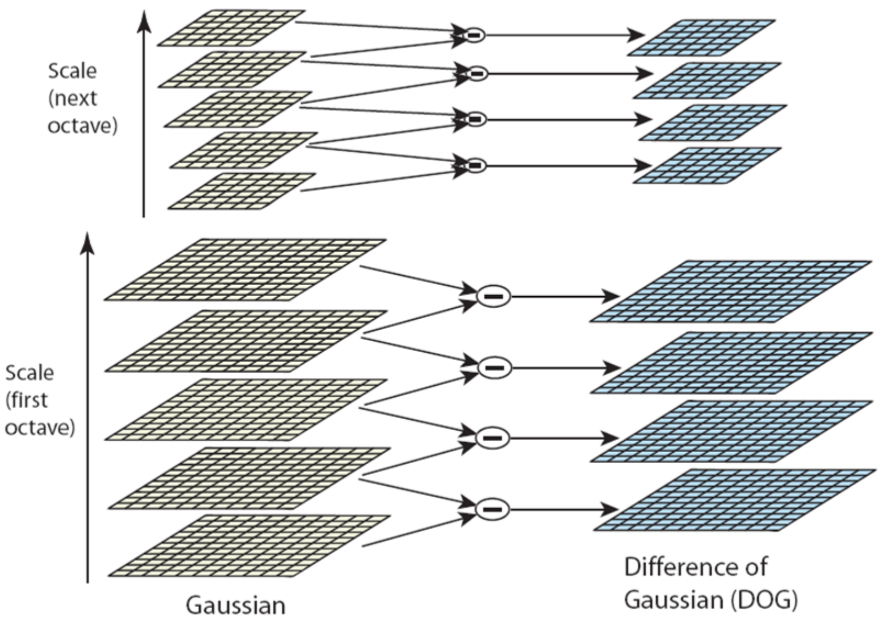
Local Features

- Capture points of interest
  - Example: SIFT, SURF, …
  - Instead of global description
- Cp. Ferrari driving video
  - House moves over different frames

Feature Extraction

Scale space extrema detection

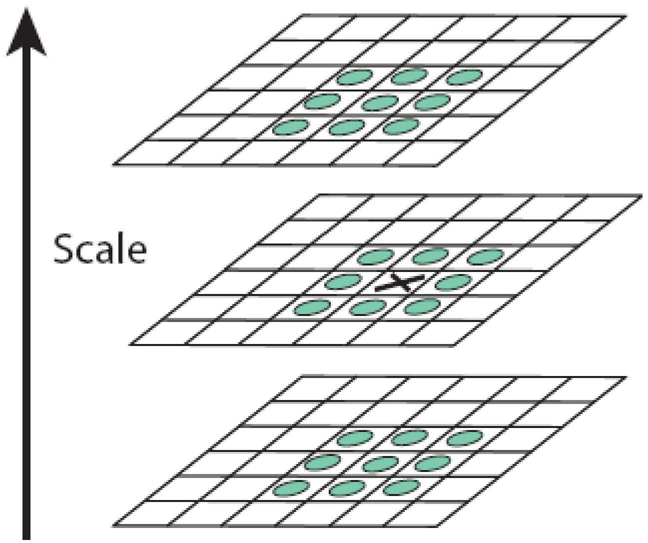
- Interest point identification
  - Difference of Gaussians
    - Use Gaussian blurred images at different octaves (resolutions)
    - Compute differences of adjacent blurred images pixel wise



Feature Extraction

Scale space extrema detection

- Compare each pixel
  - 8 direct neighbours
  - 2x9 neighbours in different scales
- Find minima and maxima
- Which are considered candidate interest points



Feature Extraction

- Scale space extrema detection produces too many candidate interest points
- I.e. SIFT reduces by
  – discarding low-contrast keypoints
  – eliminating edge responses

src. Wikipedia http://en.wikipedia.org/wiki/File:Sift_keypoints_filtering.jpg

Feature Extraction

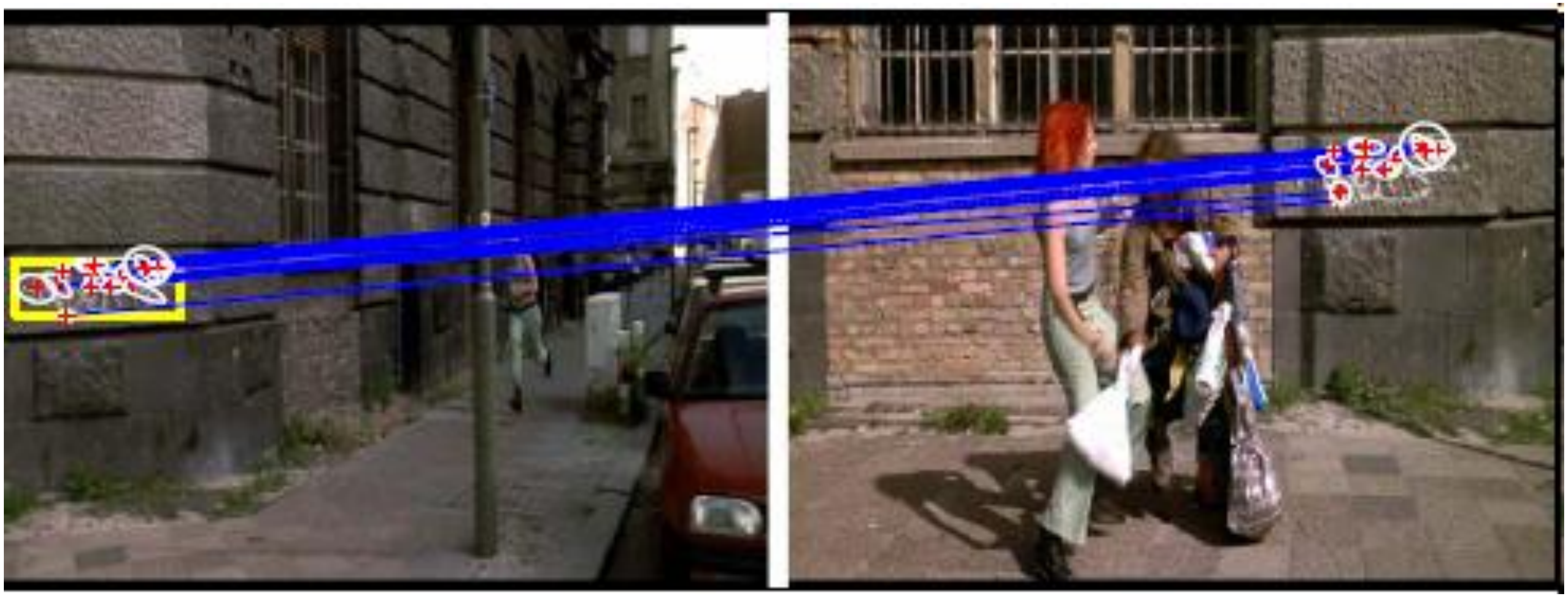
- Orientation assignment
  - based on local image gradient directions
  - achieves invariance against rotation
- Extraction
  - gradient magnitude at every scale
  - for all neighbouring pixels
  - gradient histogram with 36 bins
  - peaks are interpreted as main directions

Keypoint Descriptor

- Extracted from
  - scale of the keypoint
  - a 16x16 pixel neighborhood
  - gradient and orientation histograms
- Descriptor has 128 dimensions

Local Feature Matching

- Descriptors matching with L1, L2



Src. Sivic & Zisserman: Video Google: A Text Retrieval Approach to Object Matching in Videos, ICCV 2003, IEEE

Use Cases

- Image Stitching
  - creating panoramas from multiple images.
- 3D scene reconstruction
  - cp. Microsoft Photosynth
  - see http://photosynth.net/

Local Features

- Scale Invariant Feature Transform: SIFT
  - Lowe, David G. (1999). "Object recognition from local scale-invariant features". Proceedings of the ICCV 1999, pp. 1150–1157
- Speeded Up Robust Features: SURF
  - Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008
- Performance
  - Mikolajczyk, K.; Schmid, C. (2005). "A performance evaluation of local descriptors". IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10): 1615–1630
- In detail lecture book
  - Kristen Grauman and Bastian Leibe: Visual Object Recognition, Morgan Claypool, Synthesis, 2011

Local Features

- Process can be adapted to specific needs
  - interest point / blob detection
    - Harris Corner Detector
    - Laplacian of Gaussian (LoG)
    - Difference of Gaussians (DoG)
    - Fast Hessian Detector
    - Maximally stable extremal regions (MSER)
    - Adaptive and generic corner detection based on the accelerated segment test (AGAST)
    - … and many more
  - feature point description
    - SIFT, SURF, GLOH, HOG, LESH, BRISK, FREAK, …

Local Features in Java

- Java SIFT (ImageJ Plugin)
  - http://fly.mpi-cbg.de/~saalfeld/Projects/javasift.html
- jopensurf
  - http://code.google.com/p/jopensurf/
- MSER
  - Lire, net.semanticmetadata.lire.imageanalysis.mser.MSER
- OpenIMAJ
  - extensive library: http://www.openimaj.org/

Local Features in Applications

- OpenCV
  - platform independent
  - based on C
  - build with cmake
  - FAST, BRISK, FREAK, …
- http://opencv.willowgarage.com/wiki/

Agenda

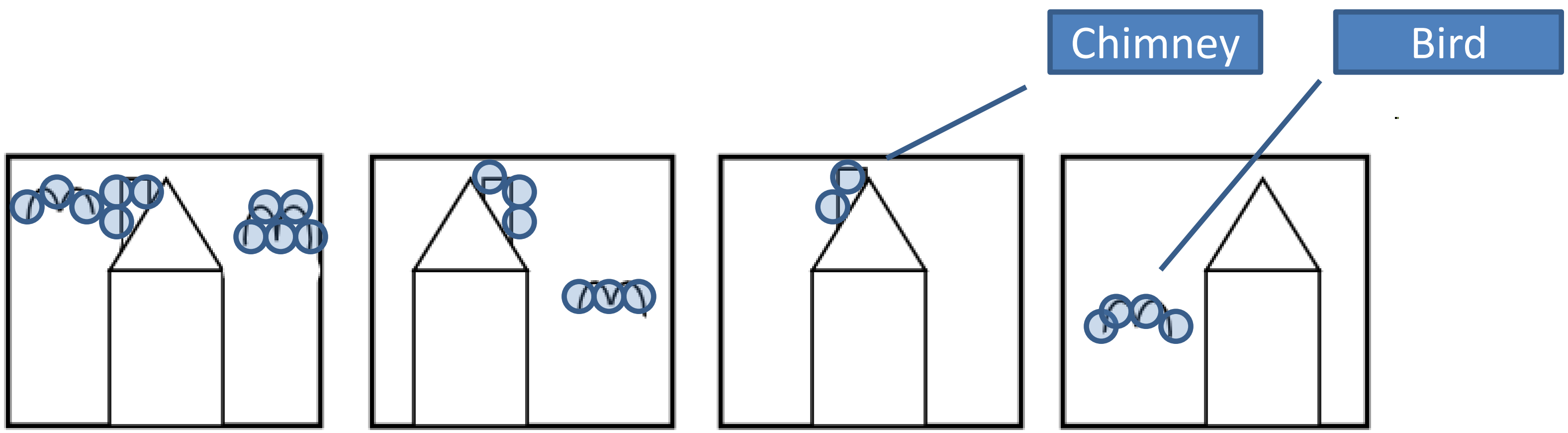
- Evaluations
- Local features
- **Bag of visual words**
- Clustering

Bag of Visual Words

- Local features are computationally expensive
  - many features per frame / image
  - pair wise distance computation leads to a huge number of distance function calls
  - e.g. *n* features vs. *m* features -> *m*n* distance function calls.

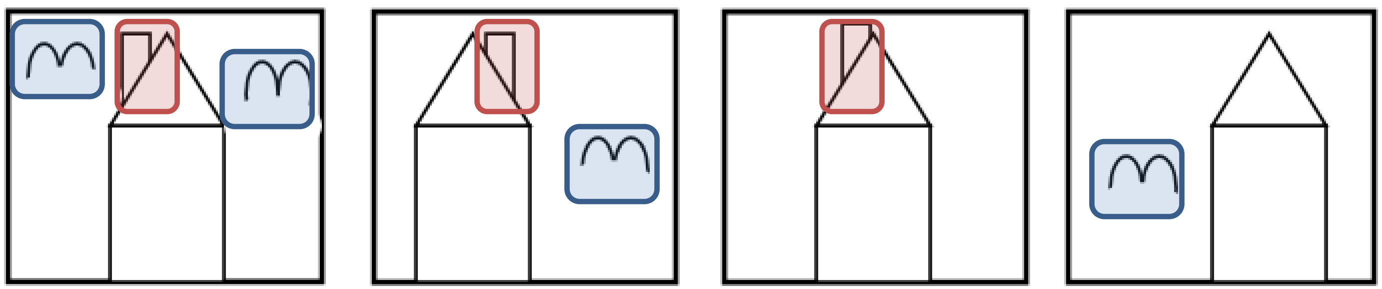
Bag of Visual Words

- Group similar local features
- Assign identifier to such a group



Bag of Visual Words

- Tag images containing features of group
  - {bird, bird, chimney}, {bird, chimney}, {chimney}, {bird}



Bag of visual words

- Groups are created unsupervised
  - not named, no semantic entities
  - model created is called visual vocabulary or codebook
- Group labels are called visual words
  - just a number, not a concept

BoVW Pipeline Overview

Local Feature Extraction → Visual Vocabulary Generation → Assignment of Visual Words

Local Feature Extraction

- Extract SIFT / SURF features
  - $k_i \gg 1$ features for image I_i
  - the bigger the image the more features

Visual Vocabulary Generation

- Select representative sample
- Cluster the union set of features
  - to a pre-selected number of clusters

- Example: 1M images
  - Select 50,000 randomly
  - Cluster features of the 50k images

Assignment of Visual Words

- For each image I in the corpus
  - For each feature of I
    - Find the best matching cluster (center)
    - Assign visual word to the image

Best practice

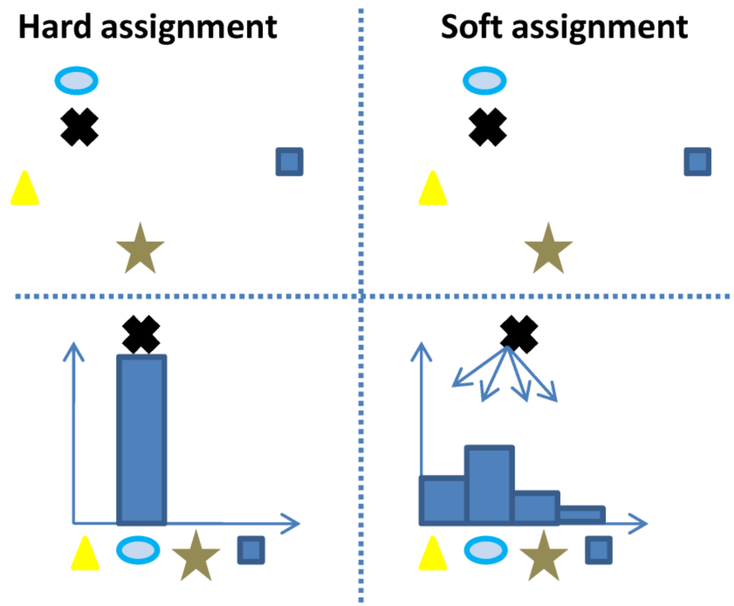
- Representative sample of documents
  - random sampling
  - up to a manageable number of features
- Vocabulary generation
  - parallel or distributed implementation
  - re-generate when necessary
- Assignment based on medians / medoids
  - employ good index structure (e.g. hashing)

Example: SURF

- Simplicity data set
  – 1000 images, 10 categories, 100 images each
- SURF features (jopensurf)
  – 98 ms / image for extraction
- Vocabulary creation
  – 400 images,
  – with ~ 92.000 features (depends on sampling)
  – 10.000 clusters, ~ 2 minutes processing time

Fuzzyness

- fuzzy instead of binary assignments
  – one feature can express multiple visual words
  – based on a fuzzy membership function
  – also called “soft assignments”



Alternative Clustering Approach

- Fuzzy C-Means
  - add a feature to more than one cluster
  - adds robustness in terms of vocabulary size

Weighting

- TF works
- IDF not so well
- Distribution?

- … this is an unresolved problem.

Agenda

- Evaluations
- Local features
- Bag of visual words
- Clustering

What is Clustering?

- Clustering is **unsupervised classification** with:
  – Maximized similarity in groups
  – Minimized similarity between groups
- Clustering creates **structure**

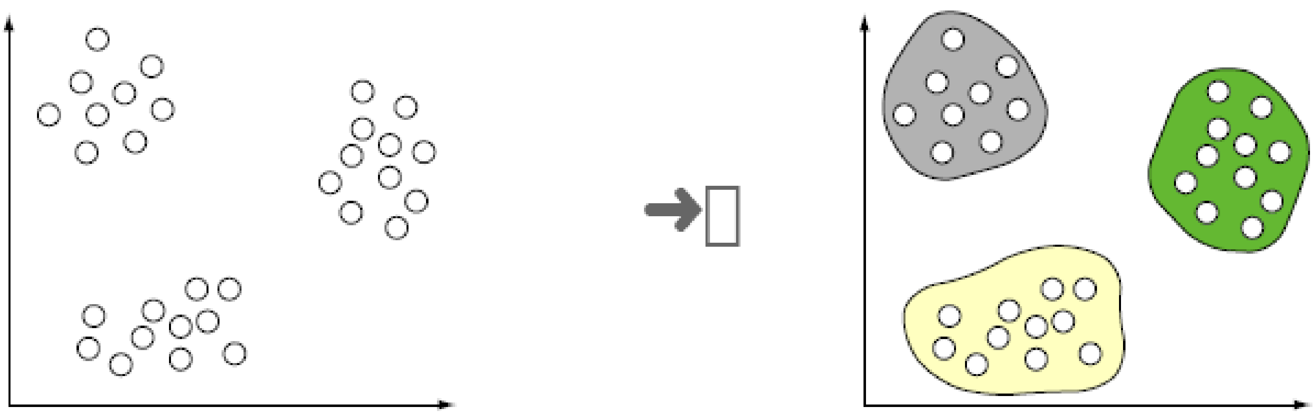
Clustering slides adapted from Benno Stein, University of Weimar
http://www.uni-weimar.de/cms/Lecture-Notes.550.0.html

and "Data Clustering: A Review", Jain, Murty & Flynn, 1999

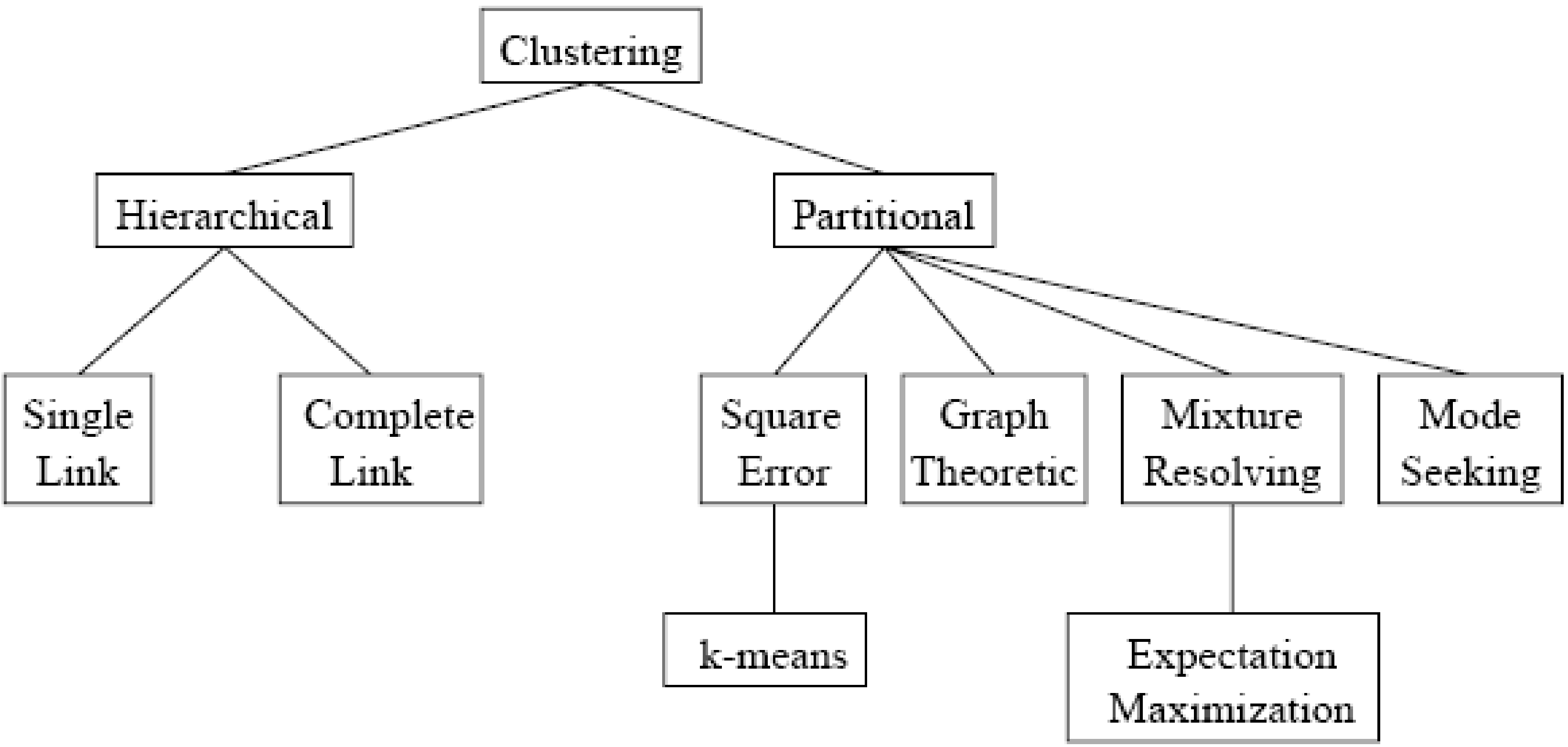
Clustering: Example

- Object has *d* features
  - – *d* ... number of dimensions

- For 2 dimensions:



Clustering Techniques

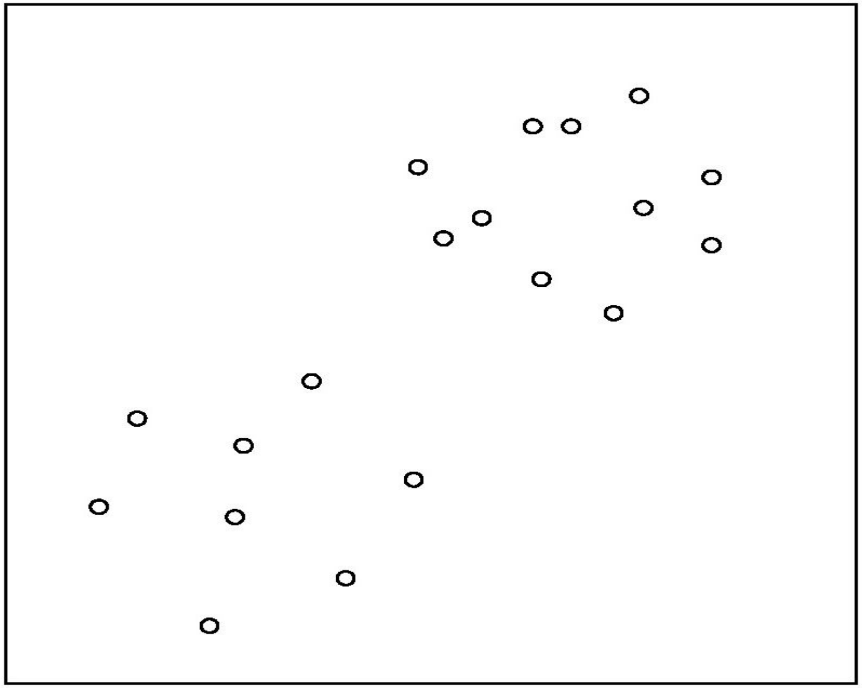


Hierarchical Clustering

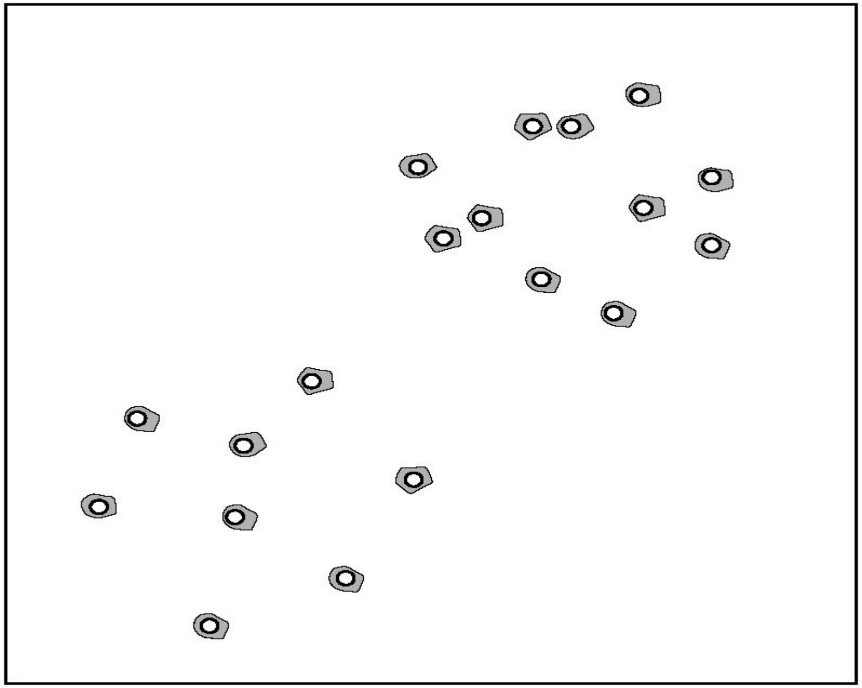
Input: $G = \langle V, E, w \rangle$. Weighted graph.
$d_{\mathcal{C}}$. Distance measure between two clusters.
Output: $T = \langle V_T, E_T \rangle$. Cluster hierarchy or dendrogram.

$\mathcal{C} = \{\{v\} \mid v \in V\}$ // define initial clustering
$V_T = \{v_C \mid C \in \mathcal{C}\}$, $E_T = \emptyset$ // define initial dendrogram
WHILE $|\mathcal{C}| > 1$ **DO**
  update_distance_matrix$(\mathcal{C}, G, d_{\mathcal{C}})$
  $\{C, C'\} = \underset{\{C_i, C_j\} \in \mathcal{C} : C_i \neq C_j}{\text{argmin}} \; d_{\mathcal{C}}(C_i, C_j)$
  $\mathcal{C} = (\mathcal{C} \setminus \{C, C'\}) \; \cup \; \{C \cup C'\}$ // clustering
  $V_T = V_T \cup \{v_{C,C'}\}$, $E_T = E_T \cup \{\{v_{C,C'}, v_C\}, \{v_{C,C'}, v_{C'}\}\}$ // dendrogram
ENDDO
RETURN(T)

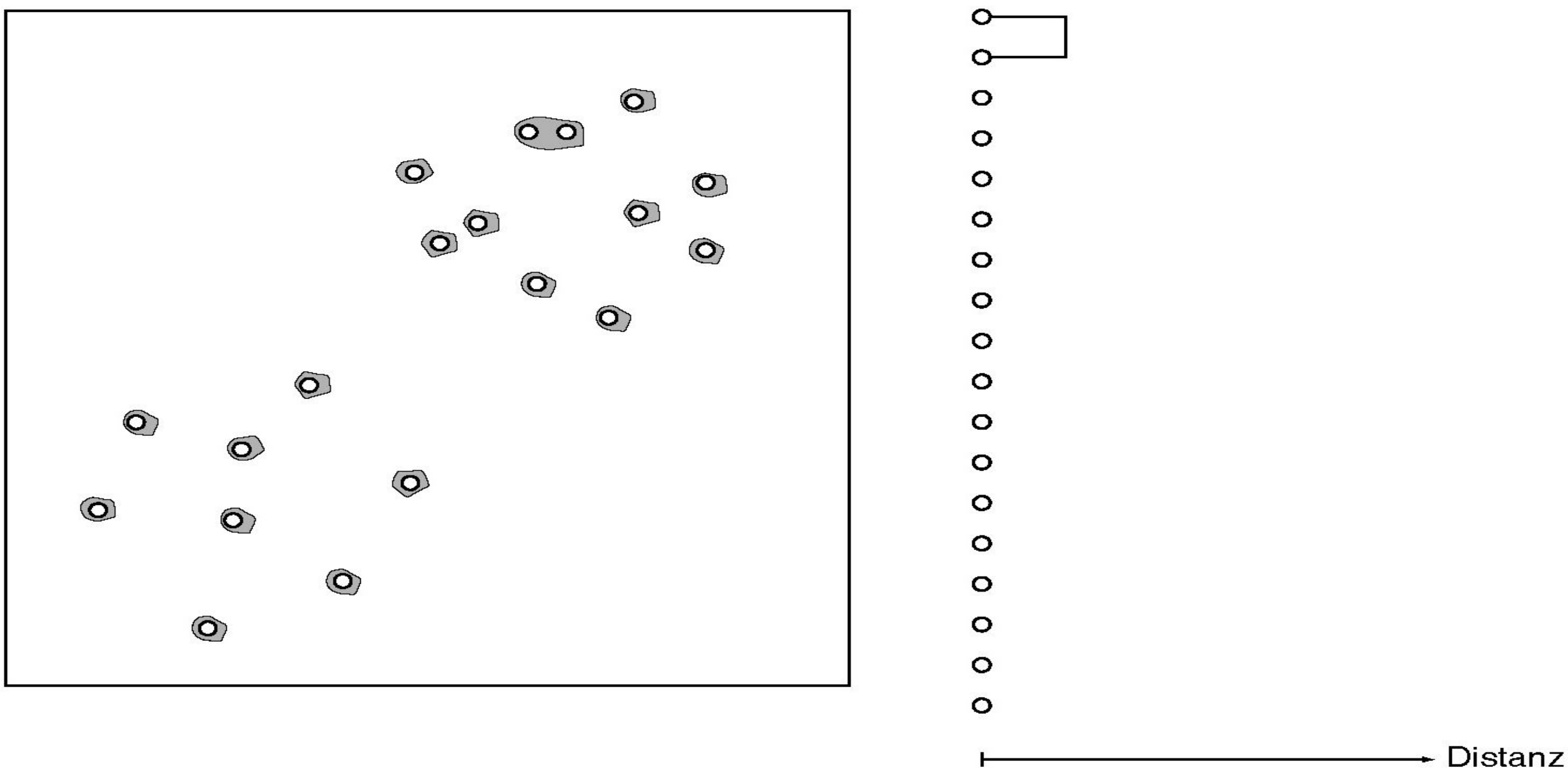
HAC: Example



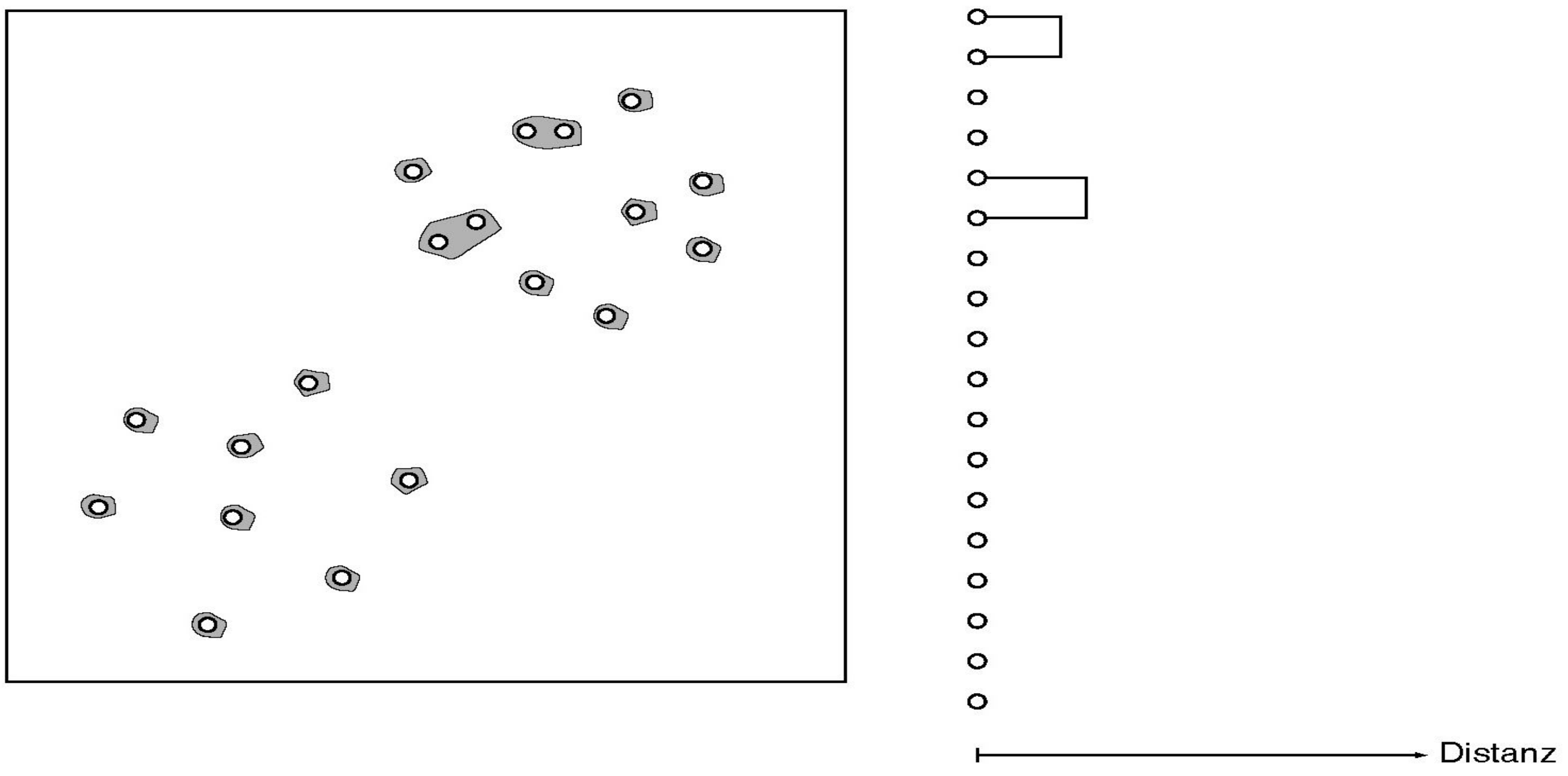
HAC: Example



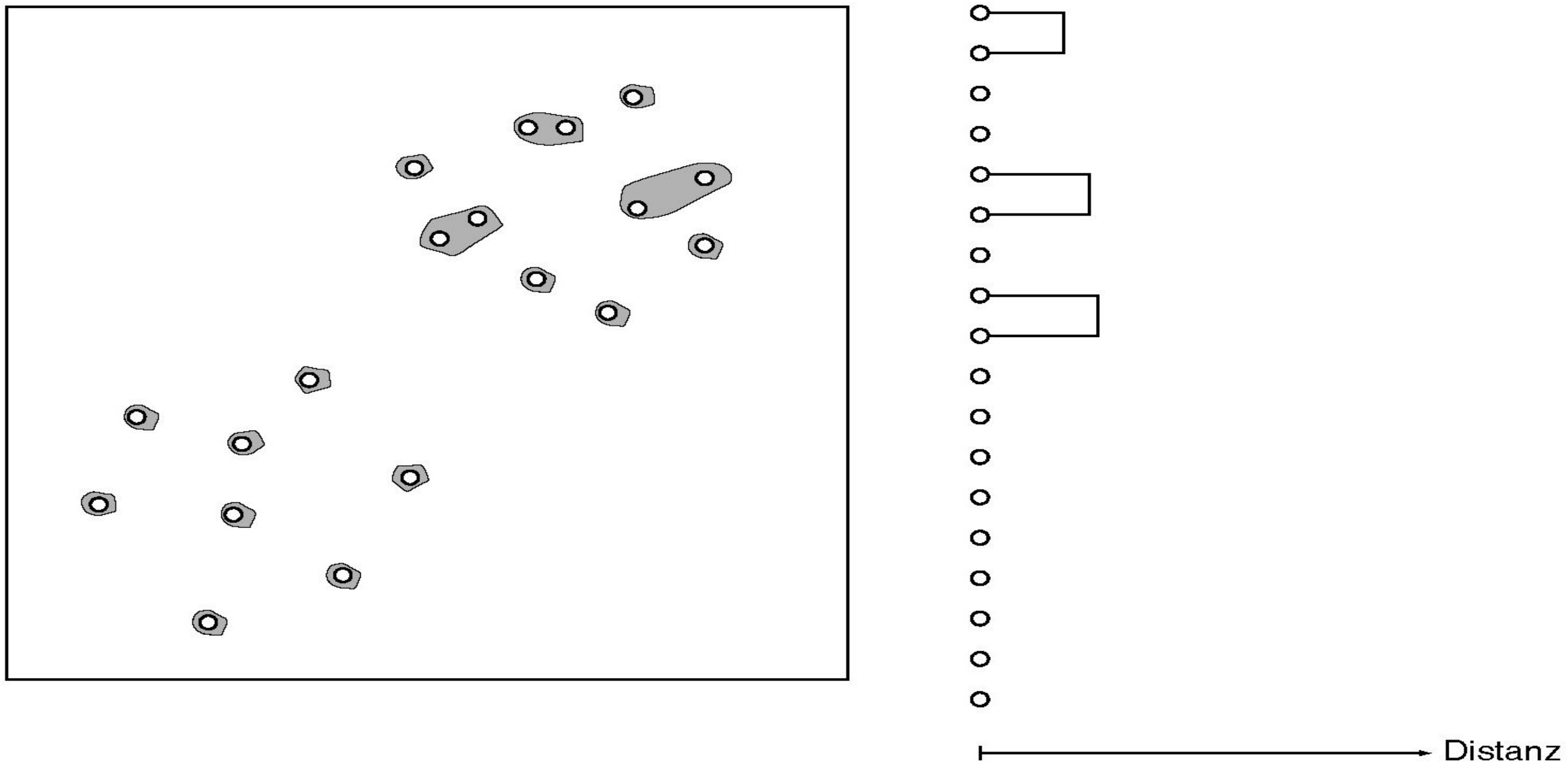
HAC: Example



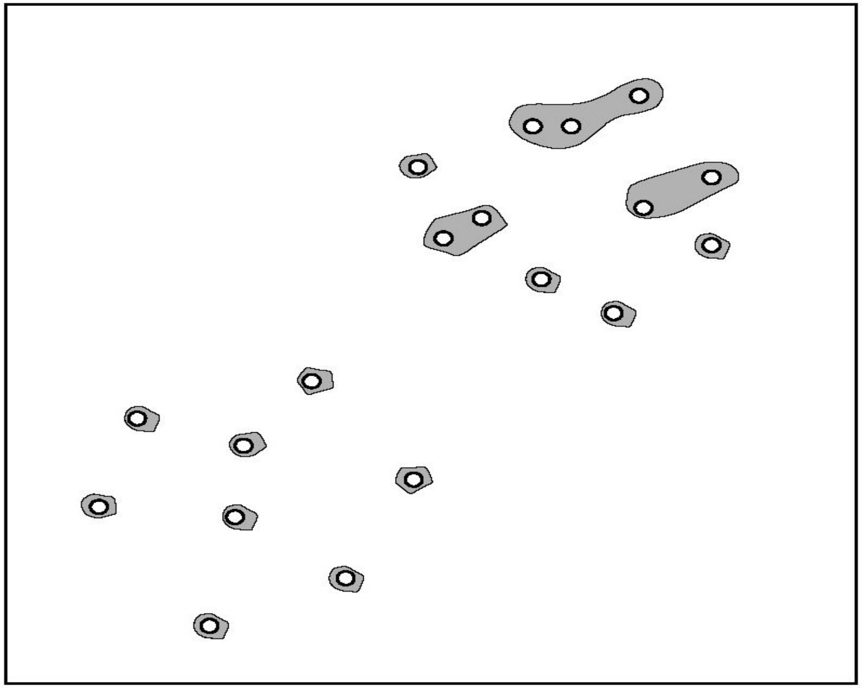
HAC: Example

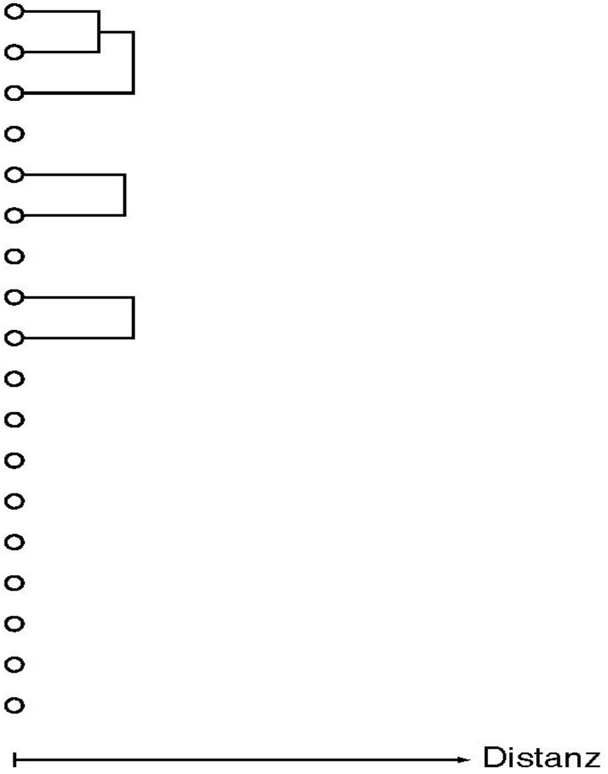


HAC: Example

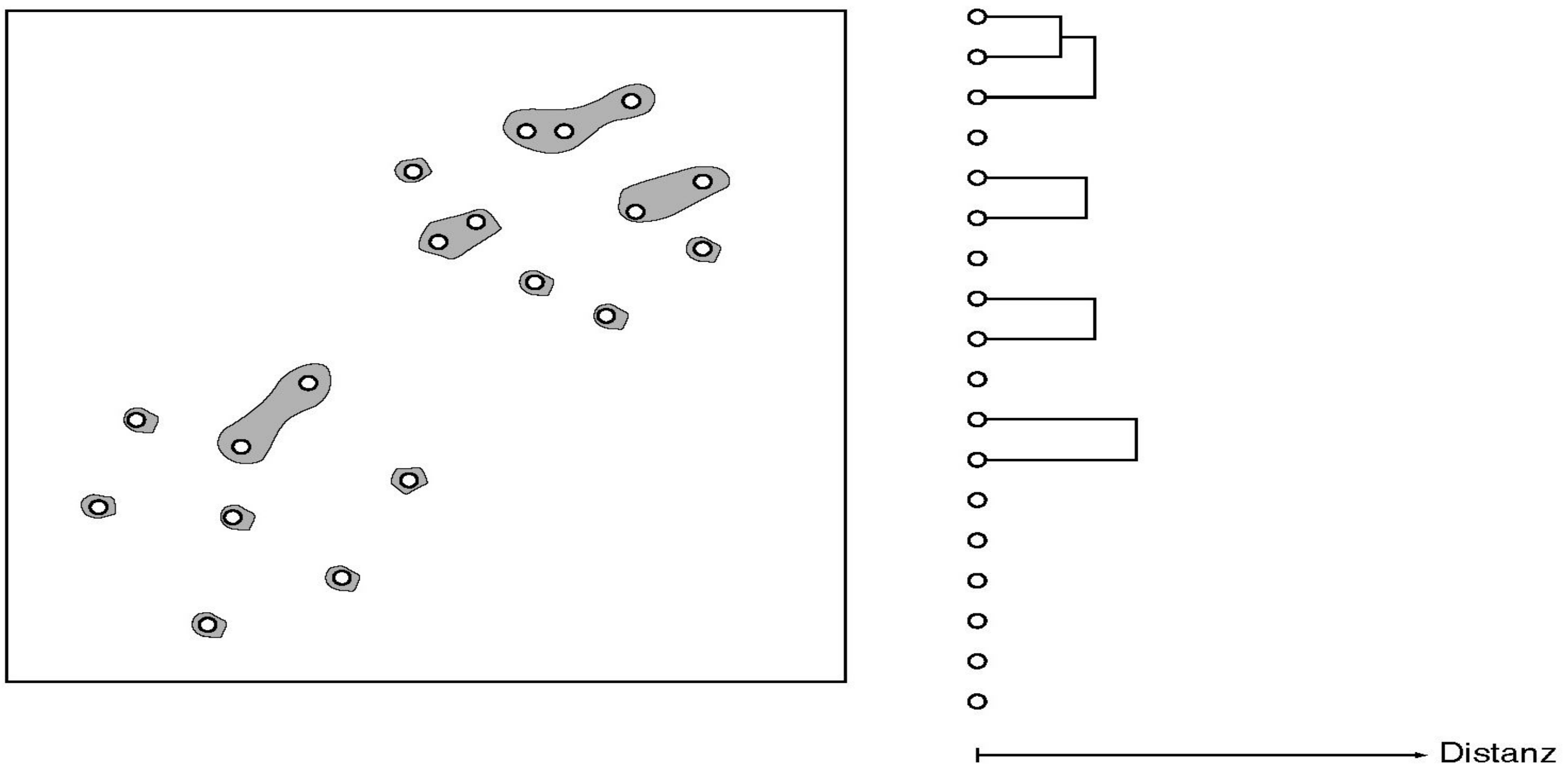


HAC: Example

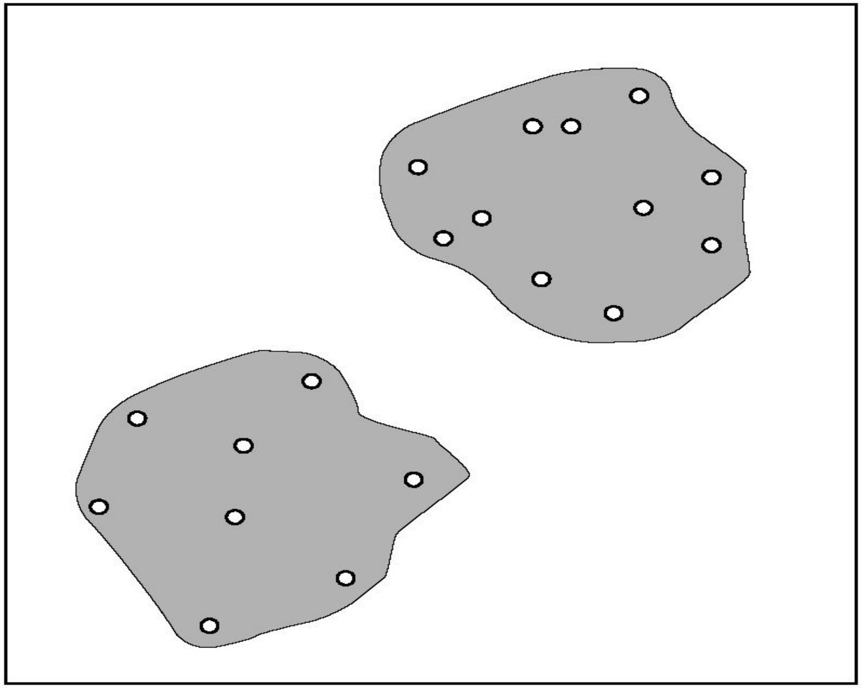


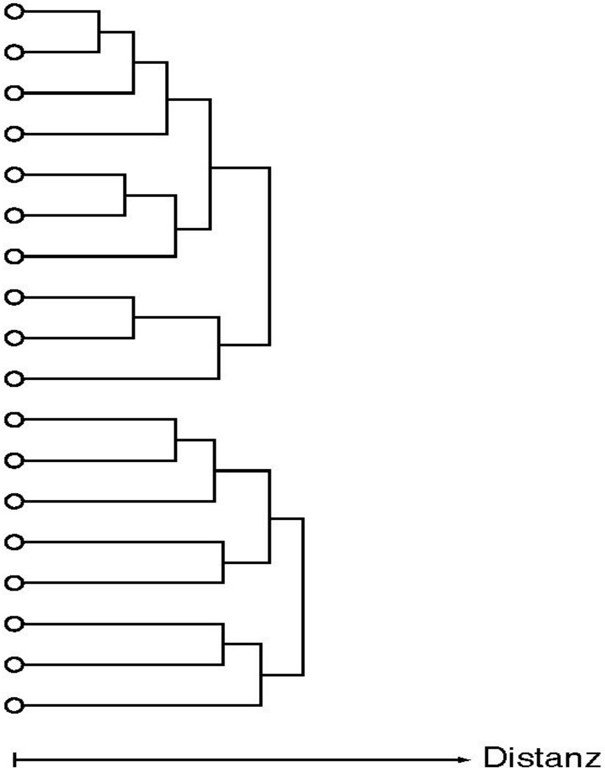


HAC: Example

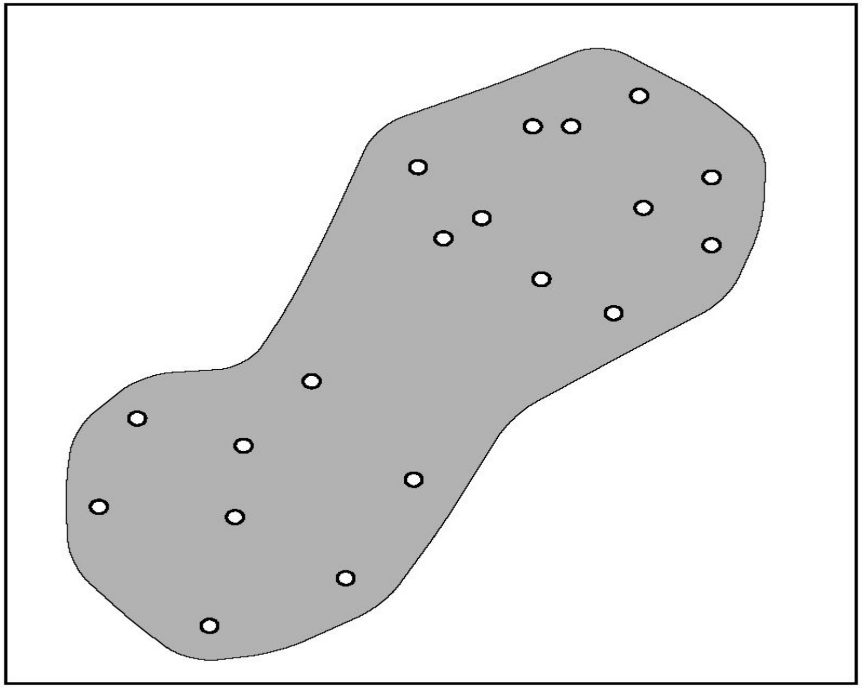


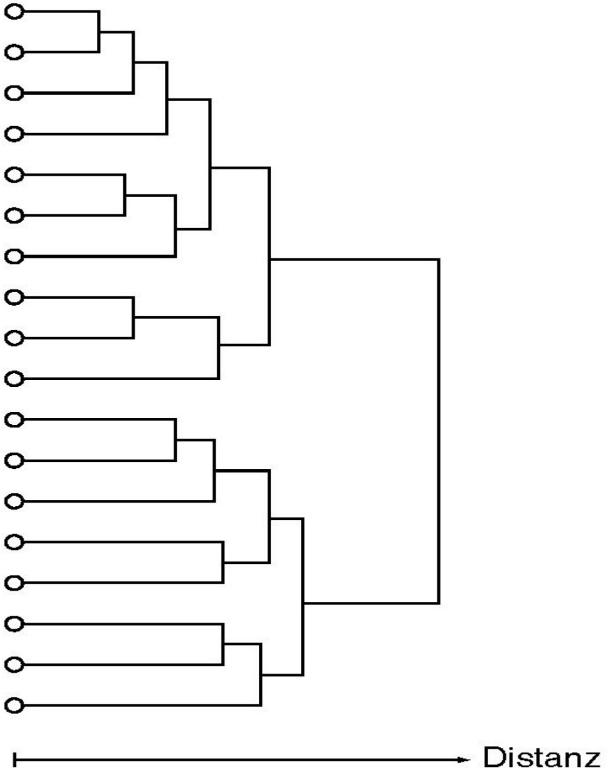
HAC: Example





HAC: Example





Cluster Distance

$$d_{\mathcal{C}}(C, C') = \min_{\substack{u \in C \\ v \in C'}} d(u, v)$$

Single-Link
(Nearest-Neighbor)

$$d_{\mathcal{C}}(C, C') = \max_{\substack{u \in C \\ v \in C'}} d(u, v)$$

Complete-Link
(Furthest-Neighbor)

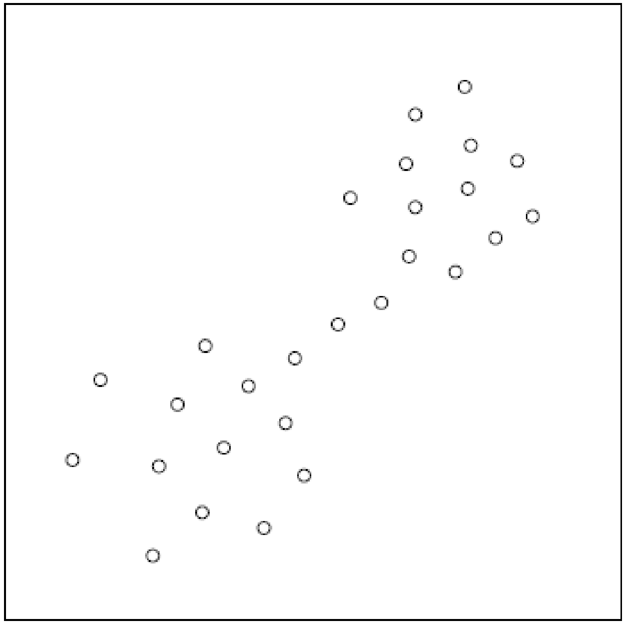
$$d_{\mathcal{C}}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{\substack{u \in C \\ v \in C'}} d(u, v)$$

(Group-)Average-Link

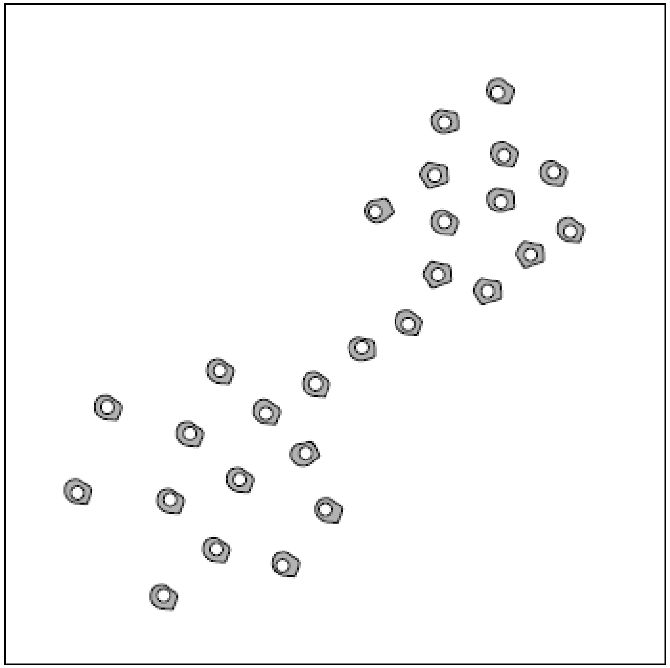
$$d_{\mathcal{C}}(C, C') = \sqrt{\frac{2 \cdot |C| \cdot |C'|}{|C| + |C'|}} \cdot ||\bar{u} - \bar{v}||$$

Ward (Varianz)

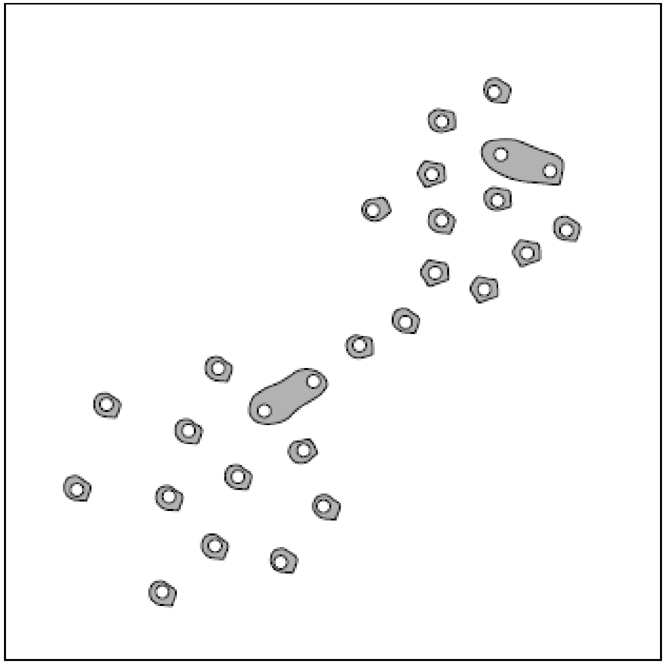
Single Link Problem: Chaining



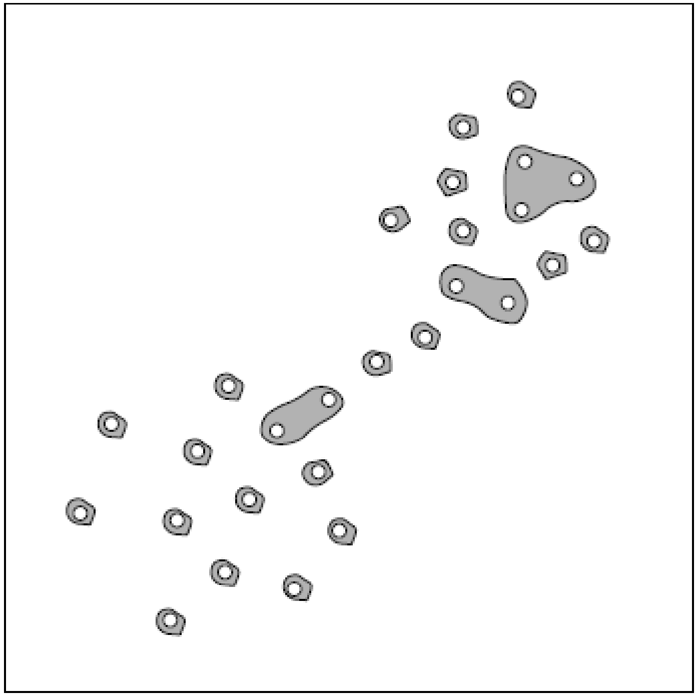
Single Link Problem: Chaining



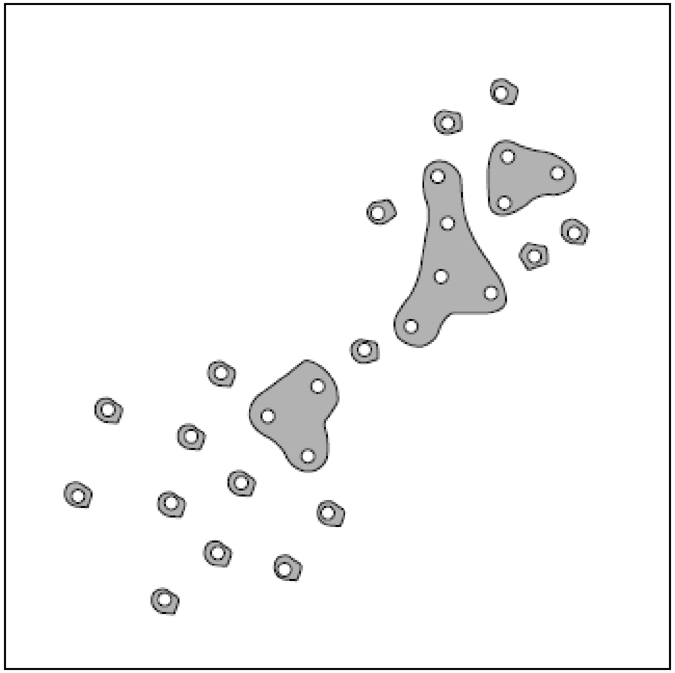
Single Link Problem: Chaining



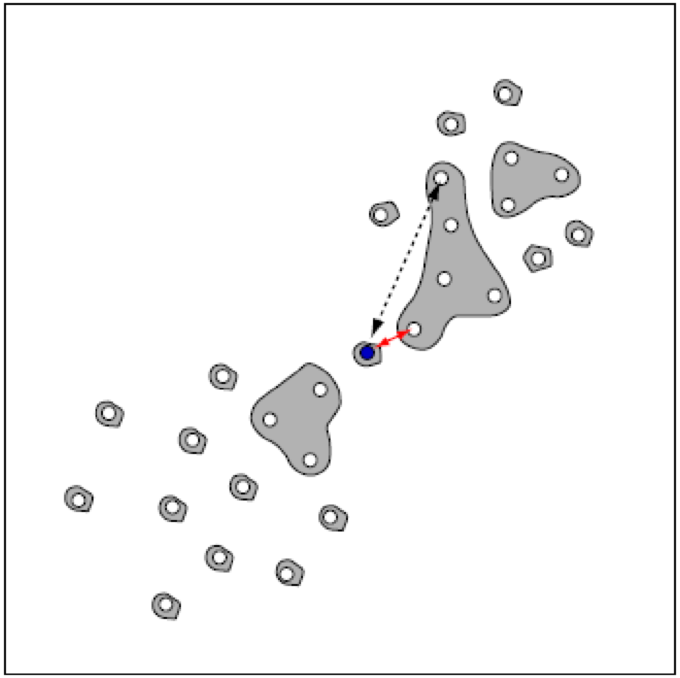
Single Link Problem: Chaining



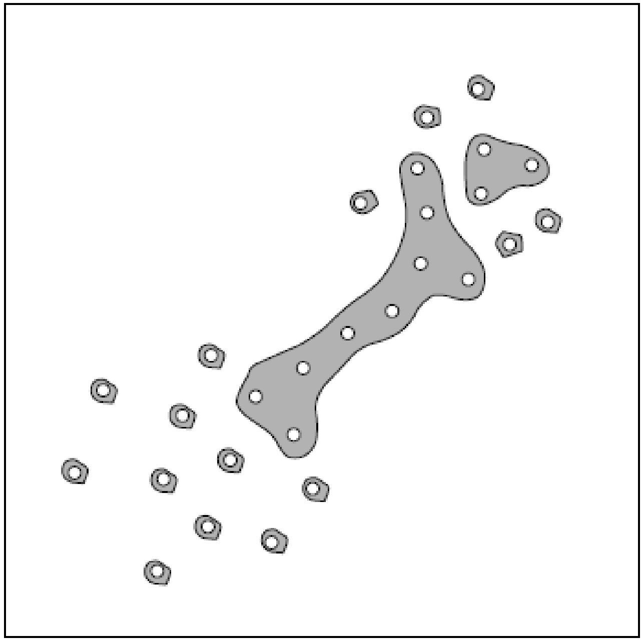
Single Link Problem: Chaining



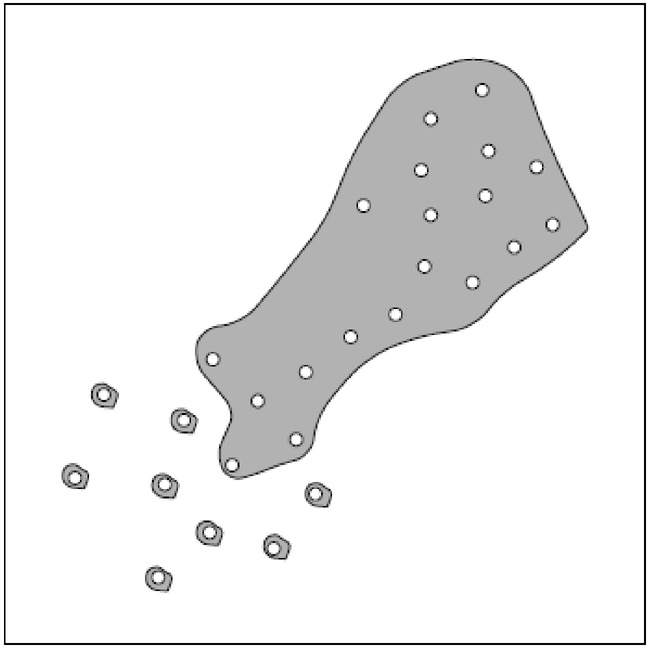
Single Link Problem: Chaining



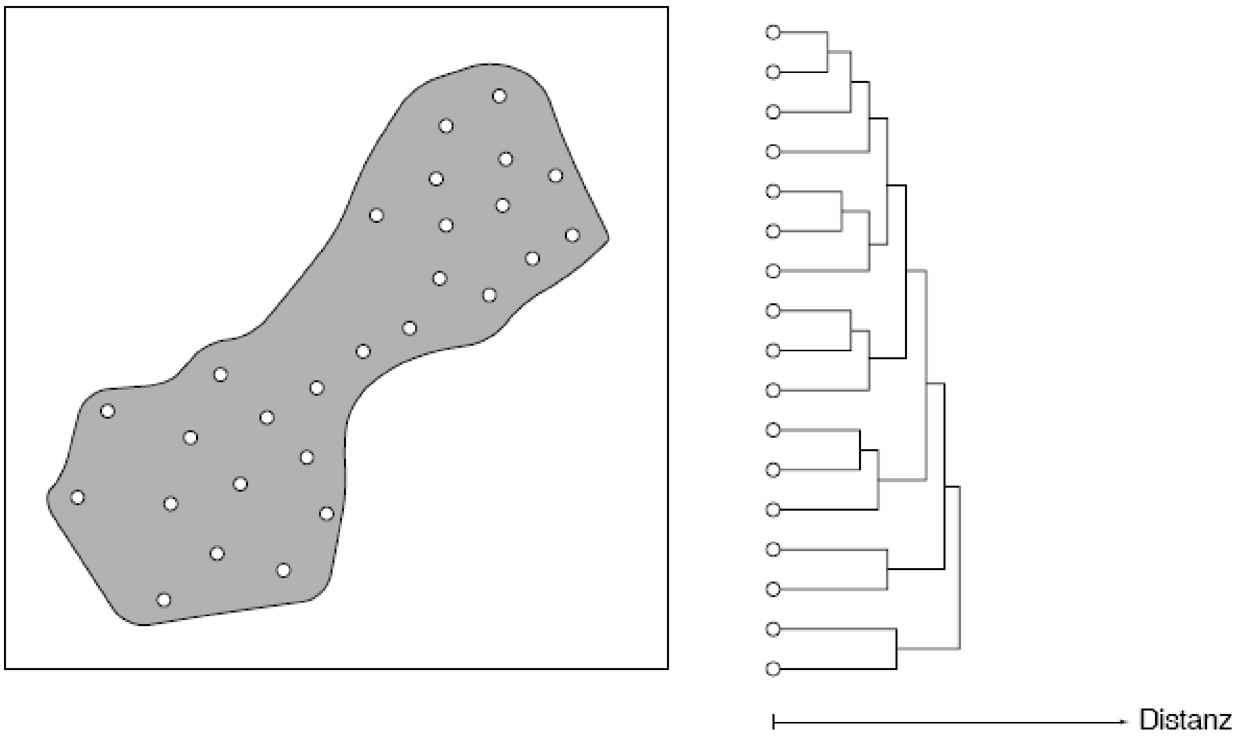
Single Link Problem: Chaining



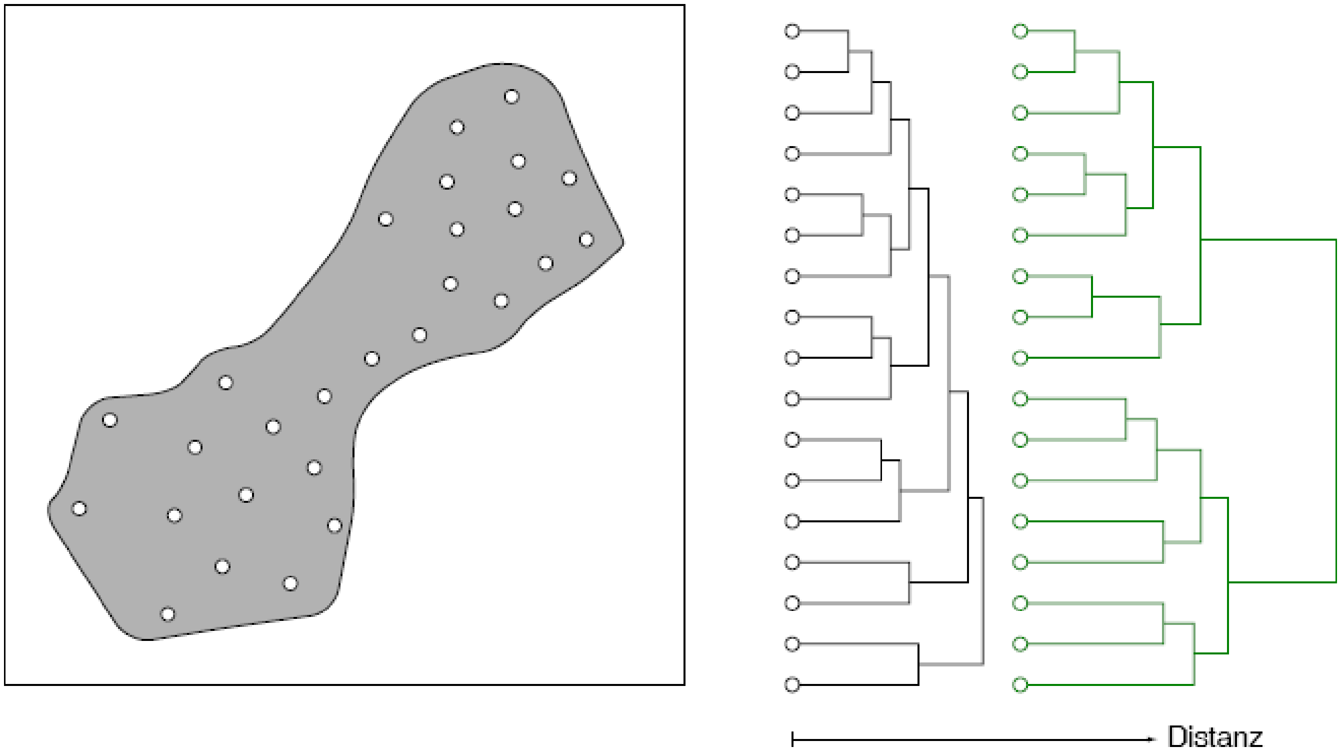
Single Link Problem: Chaining



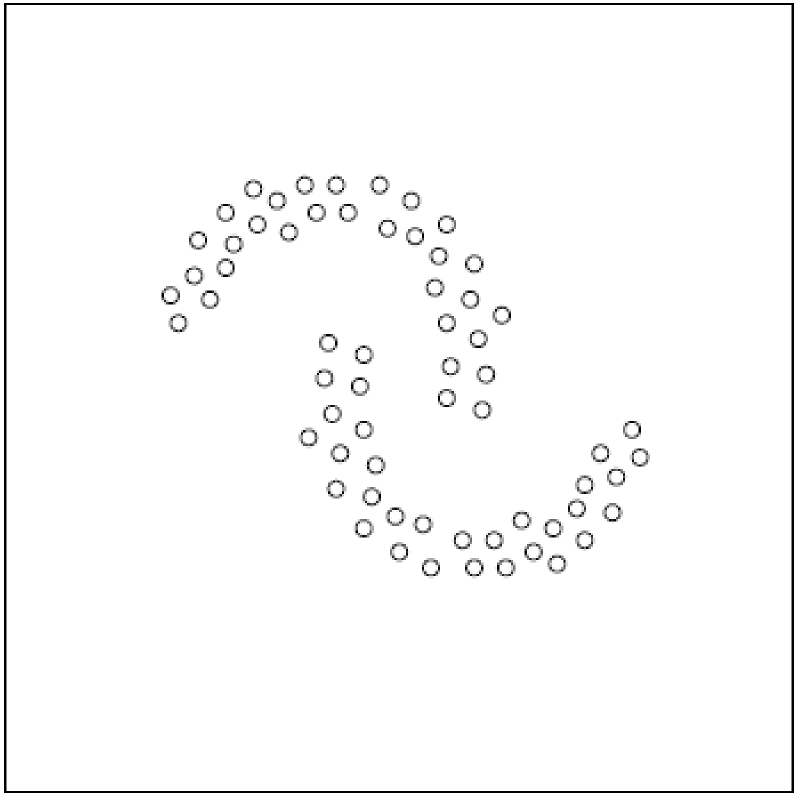
Single Link Problem: Chaining



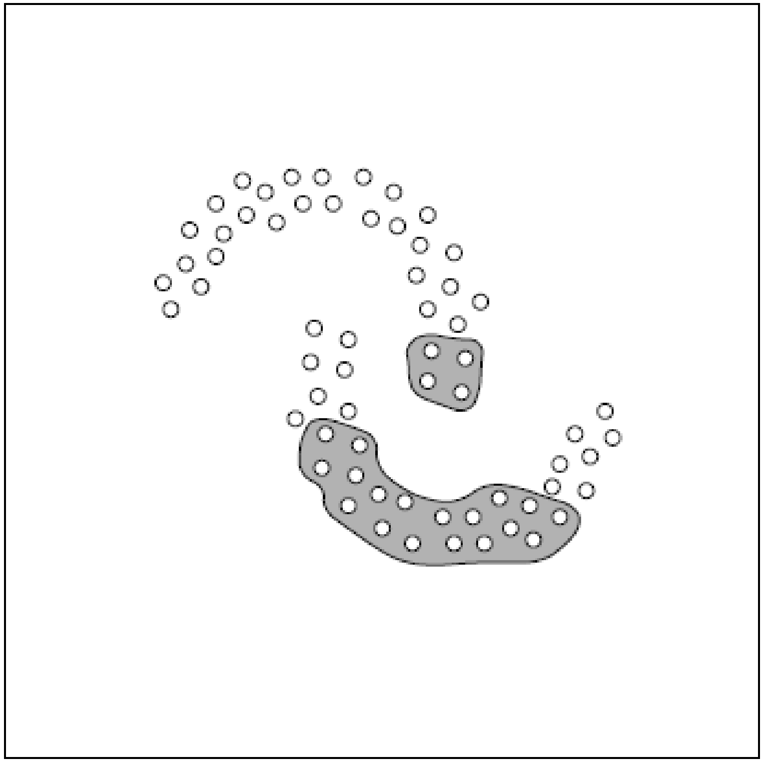
Single Link Problem: Chaining



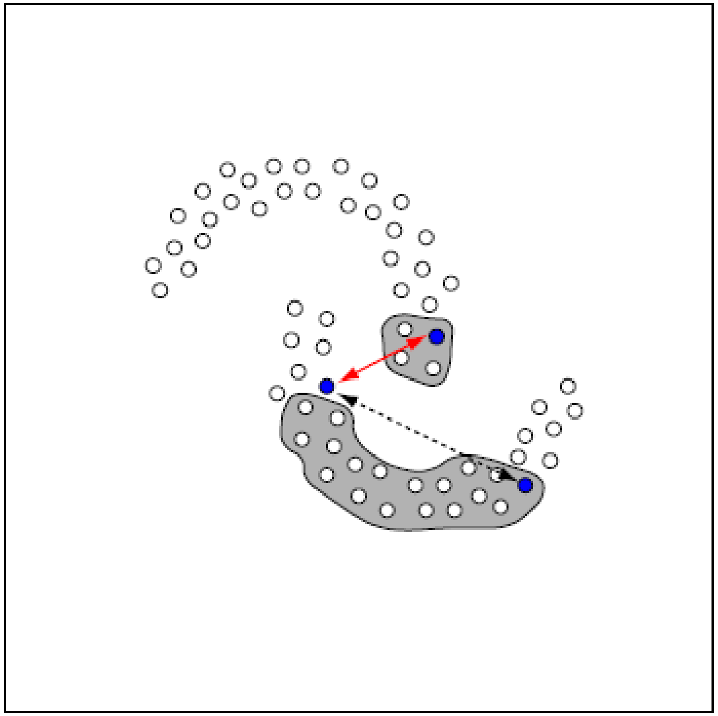
Complete Link Problem: Overlap



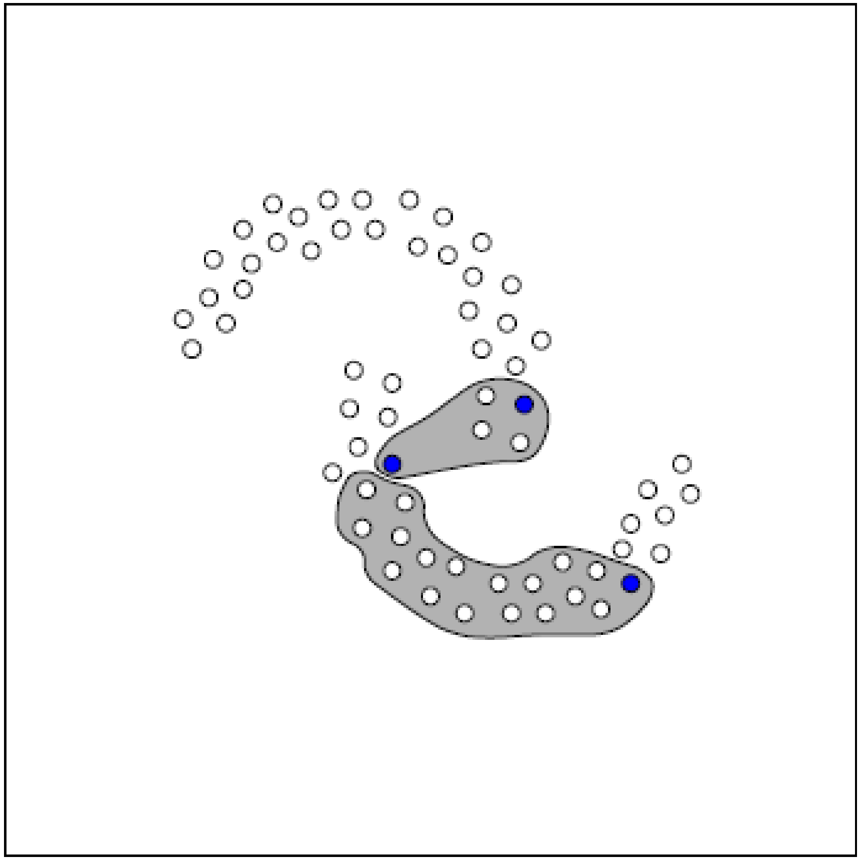
Complete Link Problem: Overlap



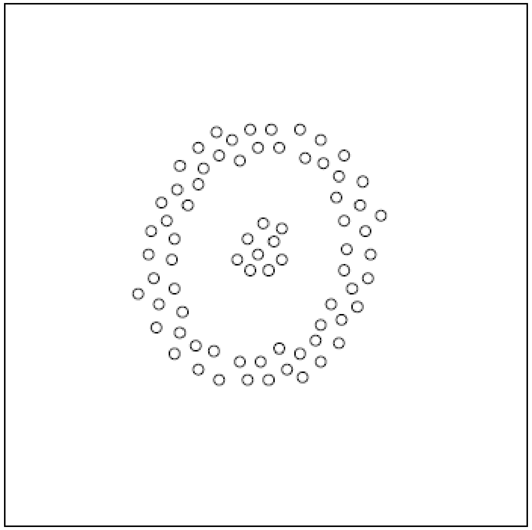
Complete Link Problem: Overlap



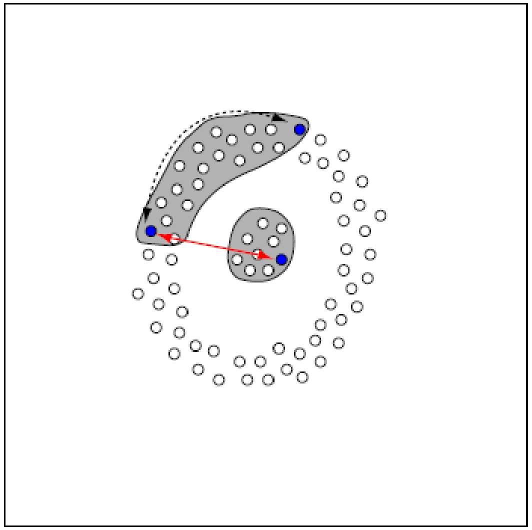
Complete Link Problem: Overlap



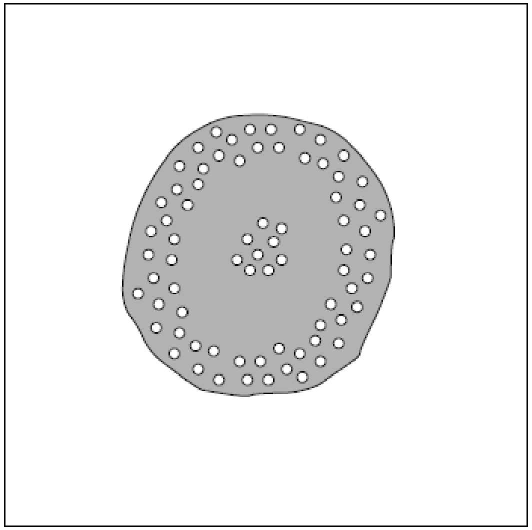
Complete Link Problem: Overlap



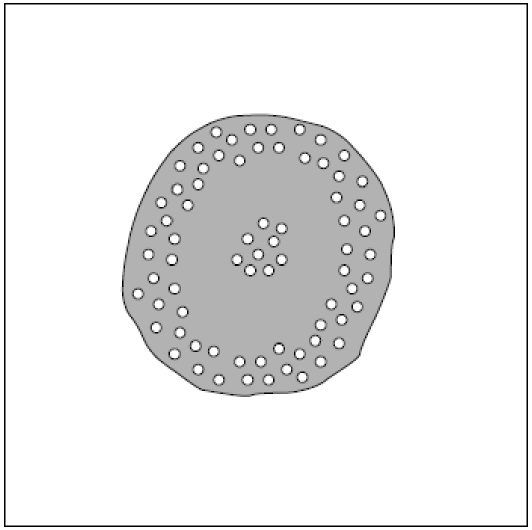
Complete Link Problem: Overlap

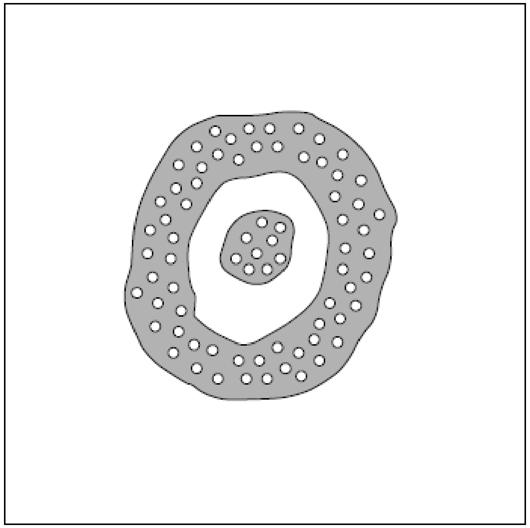


Complete Link Problem: Overlap



Complete Link Problem: Overlap





Hierarchical Clustering: Comparison

	Single Link	Complete Link	Average Link	Ward
# clusters	*small*	*high*	*medium*	*medium*
cluster type	*stretched*	*small*	*compact*	*spherical*
chaining tendency	*high*	*low*	*low*	*low*
outlier detection	*high*	*very low*	*low*	*low*

Partitional Clustering

- Only one partition of the data
  - No structure (dendrogram)
- Usually based on an optimization criterion
  - Iterated until “optimal” results
  - Multiple starting points
    - e.g. initial clusters
- Benefits for large data sets
  - But number of clusters has to be known

Iterative Clustering Algorithm

Input: $G = \langle V, E, w \rangle$. Weighted graph.
d. Distance function for nodes in V.
e. Minimization criterion for cluster representatives, based on d.
k. Number of desired clusters.

Output: $r_1, \ldots, r_k$. Cluster representatives.

$t = 0$
FOR $i = 1$ to k **DO** $r_i(t) = \mathit{choose}(V)$ // init representatives
REPEAT
  FOR $i = 1$ to k **DO** $C_i = \emptyset$
  FOREACH $v \in V$ **DO** // find nearest representative (cluster)
    $x = \underset{i:\, i \in \{1,\ldots,k\}}{\text{argmin}}\ d(r_i(t), v)$, $C_x = C_x \cup \{v\}$
  ENDDO
  FOR $i = 1$ to k **DO** $r_i(t) = \mathit{minimize}(e, C_i)$ // update
UNTIL$(\forall r_i : d(r_i(t), r_i(t-1)) < \varepsilon \ \vee \ t > t_{\max})$
RETURN$(\{r_1(t), \ldots, r_k(t)\})$

Iterative Clustering Algorithm

1. Select an initial partition of the patterns with a fixed number of clusters and cluster centers.
2. Assign each object to its closest cluster center and compute the new cluster centers as the centroids of the clusters. Repeat this step until convergence is achieved, i.e., until the cluster membership is stable.
3. Merge and split clusters based on some heuristic information, optionally repeating step 2.

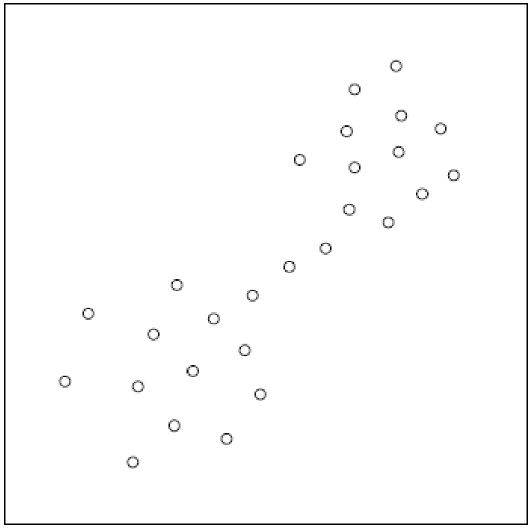
Iterative Clustering Algorithm

- Cluster representatives: Centroids (Medoids)
- Initial cluster representatives chosen randomly
- Optimization is based on the sum of squared error (distance to centroid)

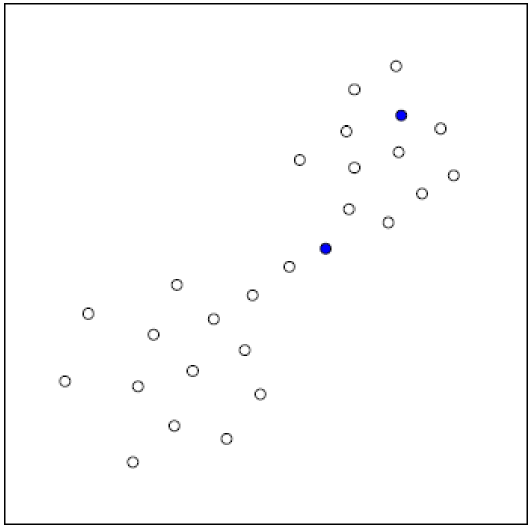
Iterative Clustering Algorithm

- Choose *k* cluster centers to coincide with *k* randomly-chosen objects or *k* randomly defined points inside the hypervolume containing the objects.
- Assign each object to the closest cluster center (centroid).
- Recompute the cluster centers (centroids) using the current cluster memberships.
- If a convergence criterion is not met, go to step 2. Typical convergence criteria are: no (or minimal) reassignment of patterns to new cluster centers, or minimal decrease in squared error

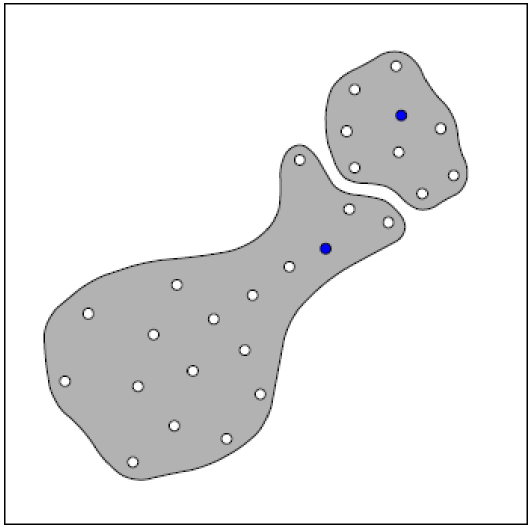
K-Means Example



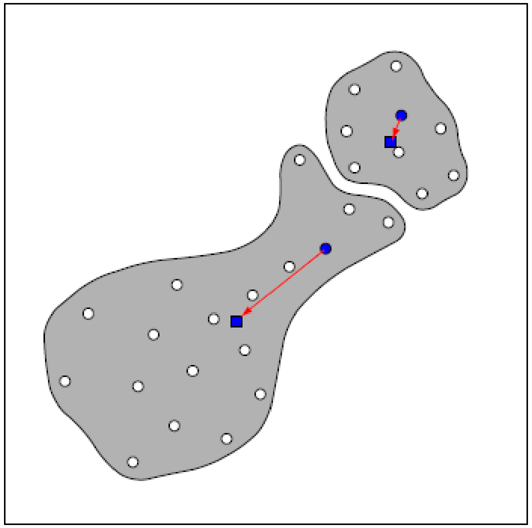
K-Means Example



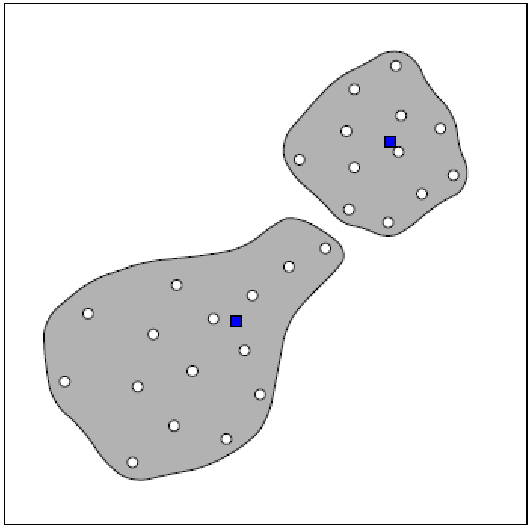
K-Means Example



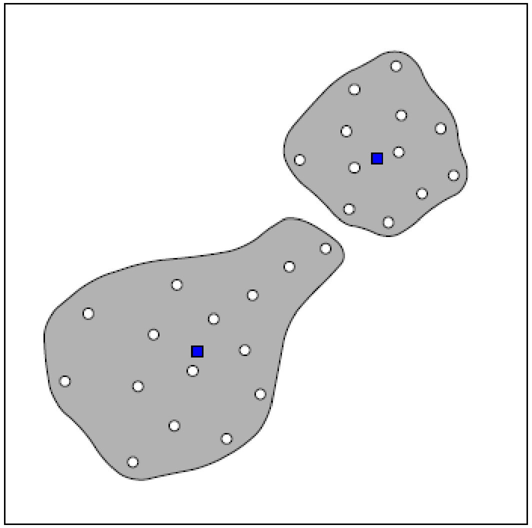
K-Means Example



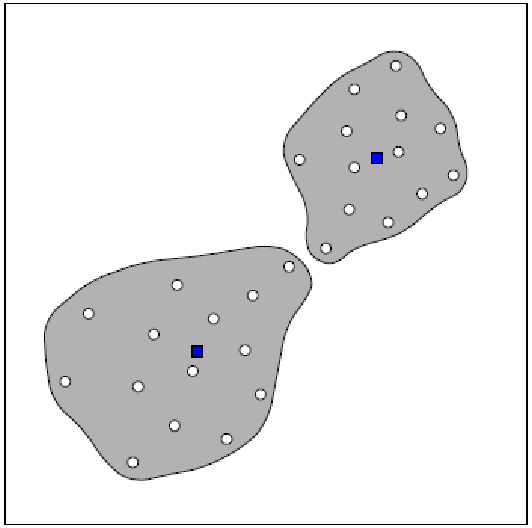
K-Means Example



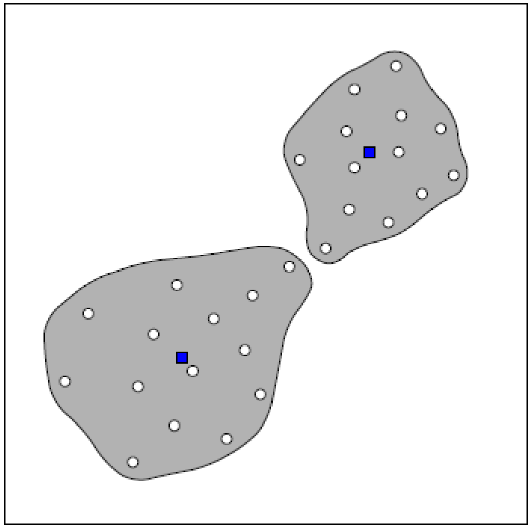
K-Means Example



K-Means Example



K-Means Example



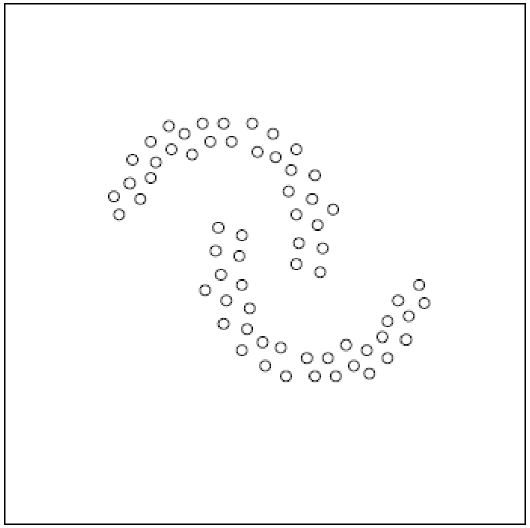
Cluster Center

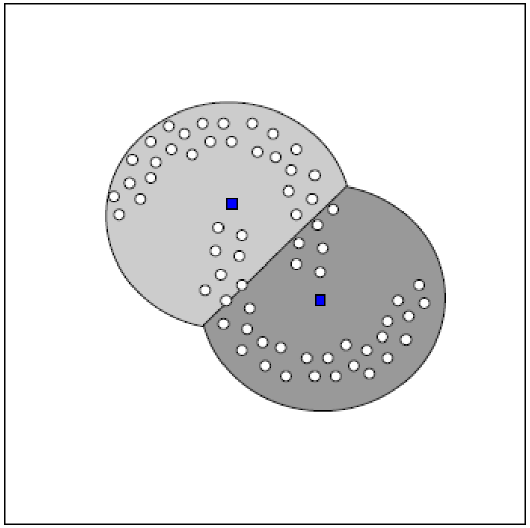
$e(\mathcal{C}) = \sum_{i=1}^{k} \sum_{v \in C_i} (v - r_i)^2$	$r_i = \bar{v}_{C_i}$	Centroid-Berechnung (k-Means)
$e(\mathcal{C}) = \sum_{i=1}^{k} \sum_{v \in C_i} \lvert v - r_i \rvert$	$r_i \in C_i$	Medoid-Berechnung (k-Medoid)
$e(\mathcal{C}) = \sum_{i=1}^{k} \max_{v \in C_i} \lvert v - r_i \rvert$	$r_i \in C_i$	k-Center
$e(\mathcal{C}) = \sum_{i=1}^{k} \sum_{v \in V} \mu_{v_i}^2 \cdot (v - r_i)^2$	$r_i = \frac{\sum_{v \in V} \mu_{v_i}^2 \cdot v}{\sum_{v \in V} \mu_{v_i}^2}$	Fuzzy-k-Means

Method Comparison

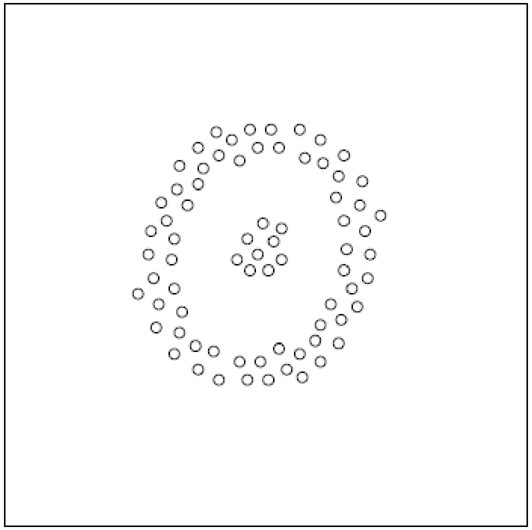
- K-Means & Fuzzy K-Means are based on interval scaled features
  - Cluster center is artificial
- K-Medoid & K-Center work with arbitrary distance and similarity functions
  - Cluster center is part of the objects
  - Medoid is more robust against outliers

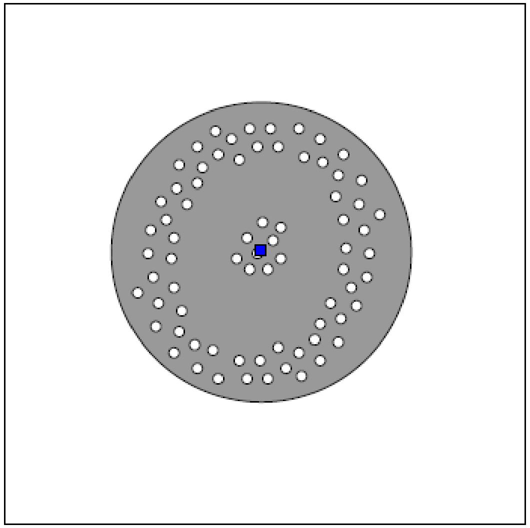
K-Means Problems





K-Means Problems





K-Means Problems