Information Retrieval
(Text Clustering)

Fabio Aiolli

http://www.math.unipd.it/~aiolli

Dipartimento di Matematica Pura ed Applicata
Università di Padova

Anno Accademico 2008/2009

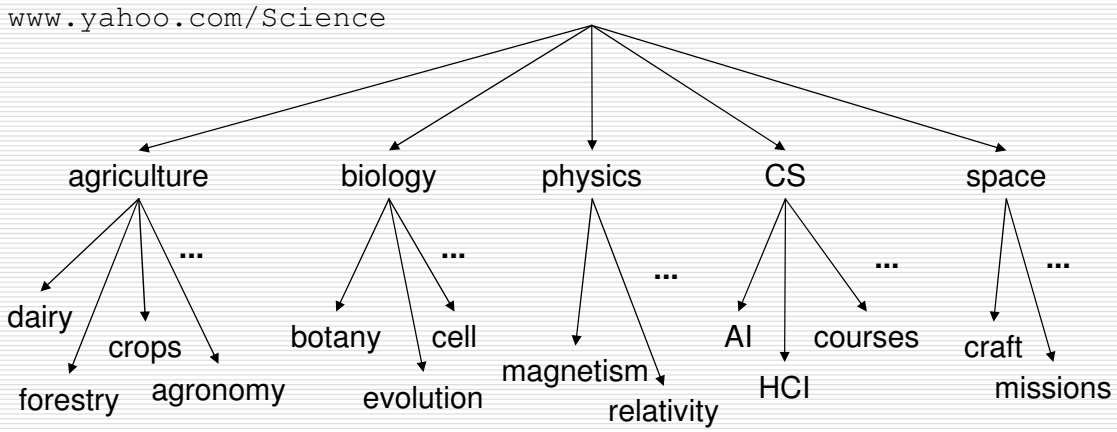
What is clustering?

- Clustering: the process of grouping a set of objects into classes of similar objects
  - The commonest form of *unsupervised learning*
    - Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples is given
  - A common and important task that finds many applications in IR and other places
- Not only Document Clustering (e.g. terms)

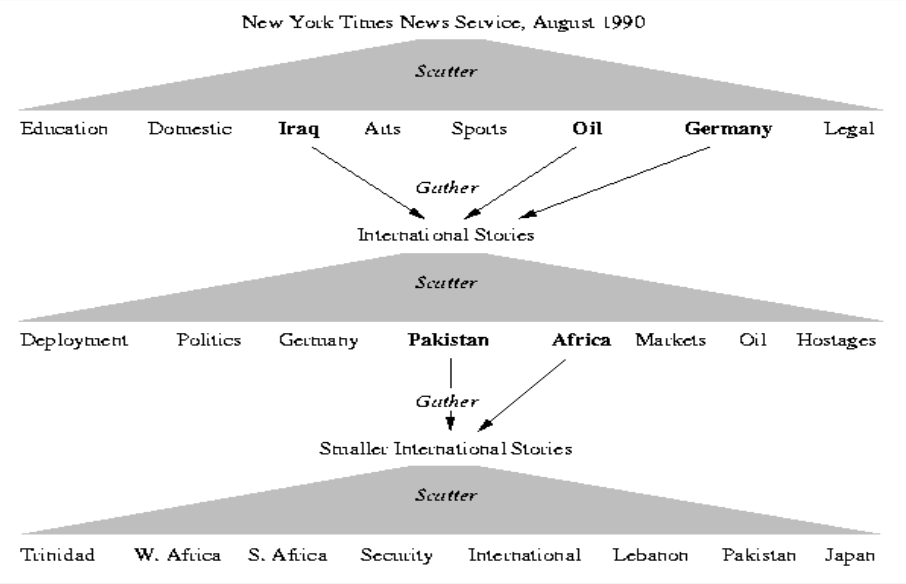
Why cluster documents?

- Whole corpus analysis/navigation
  - Better user interface
- For improving recall in search applications
  - Better search results
- For better navigation of search results
  - Effective "user recall" will be higher
- For speeding up vector space retrieval
  - Faster search

Yahoo! Hierarchy

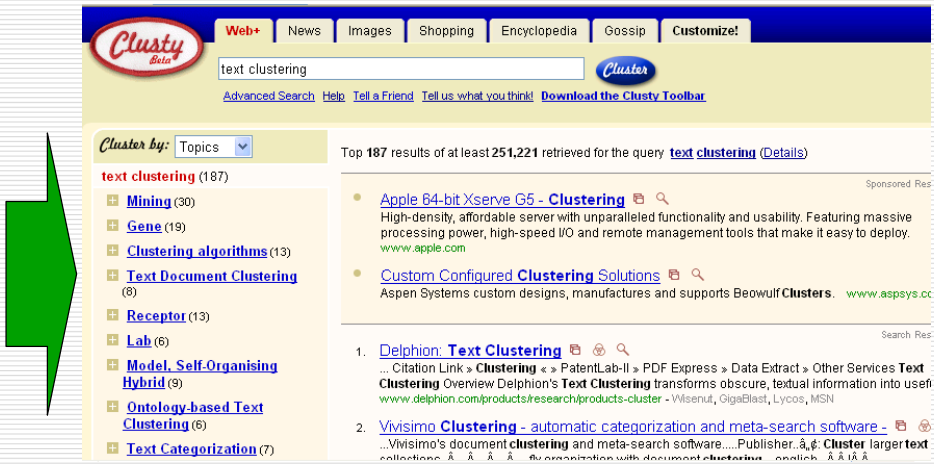


Scatter/Gather: Cutting, Karger, and Pedersen



For better navigation of search results

- For grouping search results thematically
  - clusty.com / Vivisimo



For improving search recall

- *Cluster hypothesis* - Documents with similar text are related
- Therefore, to improve search recall:
  - Cluster docs in corpus a priori
  - When a query matches a doc *D*, also return other docs in the cluster containing *D*
- Hope if we do this: The query "car" will also return docs containing *automobile*
  - Because clustering grouped together docs containing *car* with those containing *automobile.*

The Clustering Problem

Given:

- A set of documents $D=\{d_1,..d_n\}$
- A similarity measure (or distance metric)
- A partitioning criterion
- A desired number of clusters K

Compute:

- An assignment function $\gamma : D \rightarrow \{1,..,K\}$
  - None of the clusters is empty
  - Satisfies the partitioning criterion w.r.t. the similarity measure

Issues for clustering

- Representation for clustering
  - Document representation
    - Vector space? Normalization?
  - Need a notion of similarity/distance
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid "trivial" clusters - too large or small
      - In an application, if a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.

What makes docs "related"?

- Ideal: semantic similarity.
- Practical: statistical similarity
  - We will use cosine similarity.
  - Docs as vectors.
  - For many algorithms, easier to think in terms of a *distance* (rather than similarity) between docs.
  - Any kernel function can be used

Objective Functions

- Often, the goal of a clustering algorithm is to optimize an objective function
- In this cases, clustering is a search (optimization) problem
- K^N / K! different clustering available
- Most partitioning algorithms start from a guess and then refine the partition
- Many local minima in the objective function implies that different starting point may lead to very different (and unoptimal) final partitions

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the document representation and the similarity measure used

External criteria for clustering quality

- Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
- Assesses a clustering with respect to ground truth
- Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1,..,\omega_k$ with n_i members.

External Evaluation of Cluster Quality

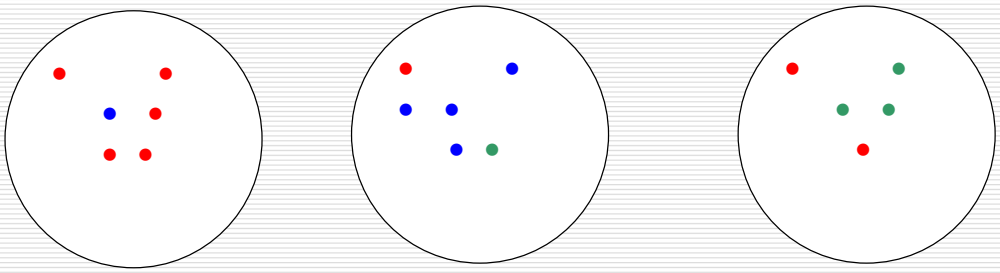
- Simple measure: purity, the ratio between the dominant class in the cluster π_i and the size of cluster ω_i

$$Purity(\omega_k) = \frac{1}{|\omega_k|} \max_j n_{kj}, \;\; n_{kj} = |\omega_k \cap c_j|$$

- Others are entropy of classes in clusters (or mutual information between classes and clusters)

$$I(\Omega, C) = \sum_k \sum_j P(\omega_k c_j) \log \frac{P(\omega_k c_j)}{P(\omega_k)P(c_j)}$$

Purity example



Cluster I Cluster II Cluster III

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

Rand Index

Number of points	Same Cluster in clustering	Different Clusters in clustering
Same class in ground truth	A (tp)	C (fn)
Different classes in ground truth	B (fp)	D (tn)

Rand index: symmetric version

$$RI = \frac{A + D}{A + B + C + D}$$

Compare with standard Precision and Recall.

$$P = \frac{A}{A + B} \qquad R = \frac{A}{A + C}$$

Rand Index example: 0.68

Number of points	Same Cluster in clustering	Different Clusters in clustering
Same class in ground truth	20	24
Different classes in ground truth	20	72

Clustering Algorithms

- Partitional algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - *K* means clustering
    - Model based clustering
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive

Partitioning Algorithms

- Partitioning method: Construct a partition of *n* documents into a set of *K* clusters
- Given: a set of documents and the number *K*
- Find: a partition of *K* clusters that optimizes the chosen partitioning criterion
  - Globally optimal: exhaustively enumerate all partitions
  - Effective heuristic methods: *K*-means and *K*-medoids algorithms

K-Means

- Assumes documents are real-valued vectors.
- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|}\sum_{\vec{x}\in c}\vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.
  - (Or one can equivalently phrase it in terms of similarities)

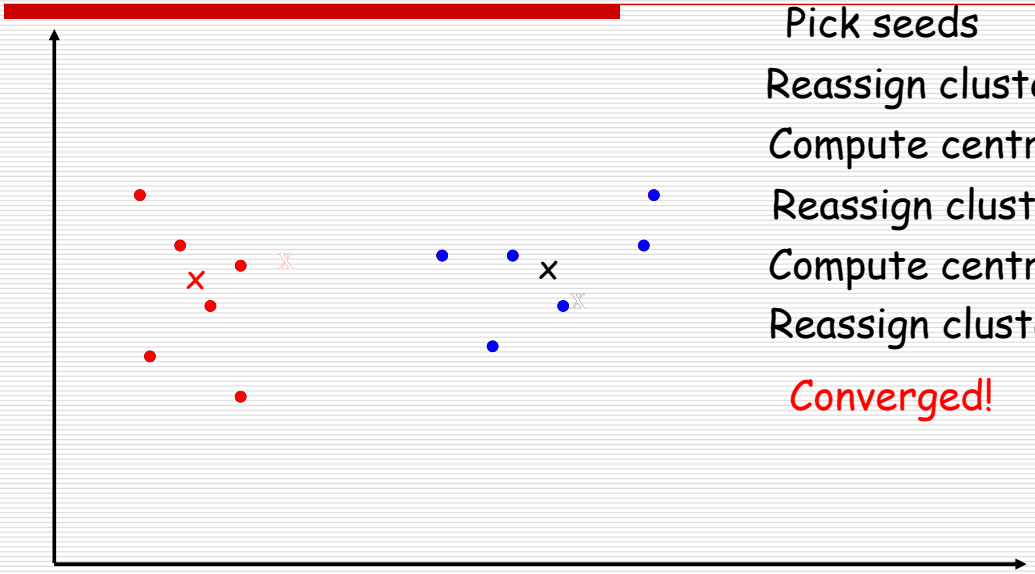
K-Means Algorithm

Select *K* random docs $\{s_1, s_2, \ldots s_K\}$ as seeds.

Until clustering converges or other stopping criterion:

1. For each doc d_i:
   Assign d_i to the cluster c_j such that $dist(x_i, s_j)$ is minimal
2. (*Update the seeds to the centroid of each cluster*)
   For each cluster c_j
   $s_j = \mu(c_j)$

K Means Example (K=2)



Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters
Converged!

Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Based on Loss function (RSS)
  - Doc partition unchanged.
  - Centroid positions don't change.

Does this mean that the docs in a cluster are unchanged?

Convergence

- Why should the *K*-means algorithm ever reach a *fixed point*?
  - A state in which clusters don't change.
- *K*-means is a special case of a general procedure known as the *Expectation Maximization (EM) algorithm*.
  - EM is known to converge.
  - Number of iterations could be large.

Convergence of *K*-Means

- Define goodness measure of cluster k as sum of squared distances from cluster centroid:
  - $G_k = \Sigma_i (d_i - c_k)^2$ (sum over all d_i in cluster k)
- $G = \Sigma_k G_k$
- Reassignment monotonically decreases G since each vector is assigned to the closest centroid.

Convergence of K-Means

- Recomputation monotonically decreases each G_k since (m_k is number of members in cluster k):
  - $\Sigma (d_i - a)^2$ reaches minimum for:
  - $\Sigma -2(d_i - a) = 0$
  - $\Sigma d_i = \Sigma a$
  - $m_K a = \Sigma d_i$
  - $a = (1/ m_k) \Sigma d_i = c_k$
- K-means typically converges quickly

Time Complexity

- Computing distance between two docs is $O(m)$ where m is the dimensionality of the vectors.
- Reassigning clusters: $O(Kn)$ distance computations, or $O(Knm)$.
- Computing centroids: Each doc gets added once to some centroid: $O(nm)$.
- Assume these two steps are each done once for I iterations: $O(IKnm)$.

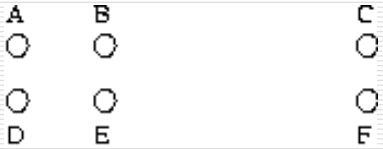
Time Complexity

- So, k-means is linear in all relevant factors (iterations, number of clusters, number of documents, and dimensionality of the space)
- But M>100.000 !!!
- Docs are sparse but centroids tend to be dense -> distance computation is time consuming
- Effective heuristics can be defined for making centroid-doc distance computation as efficient as doc-doc distance computation
- K-medoids is a variant of k-means that compute medoids (the docs closest to the centroid) instead of centroids as cluster centers.

Seed Choice

- Results can vary based on random seed selection.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
  - Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
  - Try out multiple starting points
  - Initialize with the results of another method.

Example showing sensitivity to seeds



In the above, if you start with B and E as centroids you converge to {A,B,C} and {D,E,F}
If you start with D and F you converge to {A,B,D,E} {C,F}

How Many Clusters?

- Number of clusters *K* is given
  - Partition *n* docs into predetermined number of clusters
- Finding the "right" number of clusters is part of the problem
  - Given docs, partition into an "appropriate" number of subsets.
  - E.g., for query results - ideal value of *K* not known up front - though UI may impose limits.

K not specified in advance

- Say, the results of a query.
- Solve an optimization problem
  - $MIN_K\ L(K) + \lambda\ q(K)$ which penalizes having lots of clusters
  - application dependent, e.g., compressed summary of search results list.
- Tradeoff between having more clusters (better focus within each cluster) and having too many clusters

Model-based Clustering

- A different way of posing the clustering problem is to formalize the clustering as a parameterized model Θ and then search the parameters that maximize the likelihood of the data

$$MAX\ L(D|\Theta)$$

- Where $L(D|\Theta) = \log \prod_n P(d_n|\Theta) = \sum_n \log P(d_n)$
- This can be done by an EM (Expectation Maximization procedure)
- K-means can be seen as an instance of EM when the model is a mixture of multivariate Gaussians

Model-based Clustering

- Instead of a Gaussian mixture, we focus on mixture of multivariate binomials, i.e.

$$P(d|\omega_k,\Theta) = \prod_m P(X_m=I(w_m \in d)|\omega_k)$$

- The mixture model

$$P(d|\Theta) = \sum_k \gamma_k \prod_m P(X_m=I(w_m \in d)|\omega_k)$$

- N.B. K-means performs an hard assignment while the binomial EM clustering performs a soft assignment

EM

- Maximization Step: computes the conditional parameters $q_{mk} = P(X_m = 1 \mid \omega_k)$ and the priors γ_k

$$q_{mk} = \frac{\sum_{n=1}^{N} r_{nk} I(\omega_m \in d_n)}{\sum_{n=1}^{N} r_{nk}} \qquad \gamma_k = \frac{\sum_{n=1}^{N} r_{nk}}{N}$$

- Expectation Step: computes the soft assignment of documents to clusters given the current parameters

$$r'_{nk} = \gamma_k (\prod_{w_m \in d_n} q_{mk})(\prod_{w_m \notin d_n} (1 - q_{mk}))$$

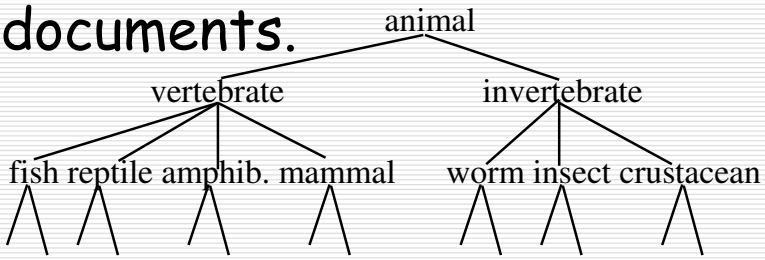
$$r_{nk} = \frac{r'_{nk}}{\sum_k r'_{nk}}$$

Considerations

- Finding good seeds is even more critical for EM than for k-means (EM is prone to get stuck in local optima)
- Therefore (as in k-means) an initial assignment is often computed by another algorithm
- If the model of the data is correct EM algorithm finds the correct structure
- Hardly a document collection can be considered generated by a simple mixture model
- At least, model based clustering allows for analysis and adaptations

Hierarchical Clustering

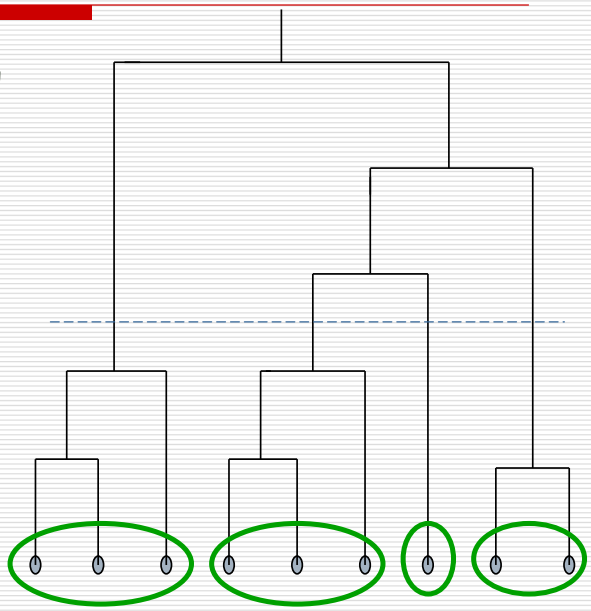
- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of documents.



- One approach: recursive application of a partitional clustering algorithm.

Dendogram: Hierarchical Clustering

- Clustering obtained by cutting the dendrogram at a desired level: each connected component forms a cluster.



The dendogram

- The y-axis of the dendogram represents the combination similarities, i.e. the similarities of the clusters merged by a the horizontal lines for a particular y
- Assumption: The merge operation is monotonic, i.e. if $s_1,..,s_{k-1}$ are successive combination similarities, then $s_1 \geq s_2 \geq ... \geq s_{k-1}$ must hold

Hierarchical Agglomerative Clustering (HAC)

- Starts with each doc in a separate cluster
  - then repeatedly joins the closest pair of clusters, until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

Closest pair of clusters

- Many variants to defining closest pair of clusters
- Single-link
  - Similarity of the *most* cosine-similar (single-link)
- Complete-link
  - Similarity of the "furthest" points, the *least* cosine-similar
- Centroid
  - Clusters whose centroids (centers of gravity) are the most cosine-similar
- Average-link
  - Average cosine between pairs of elements

Single Link Agglomerative Clustering

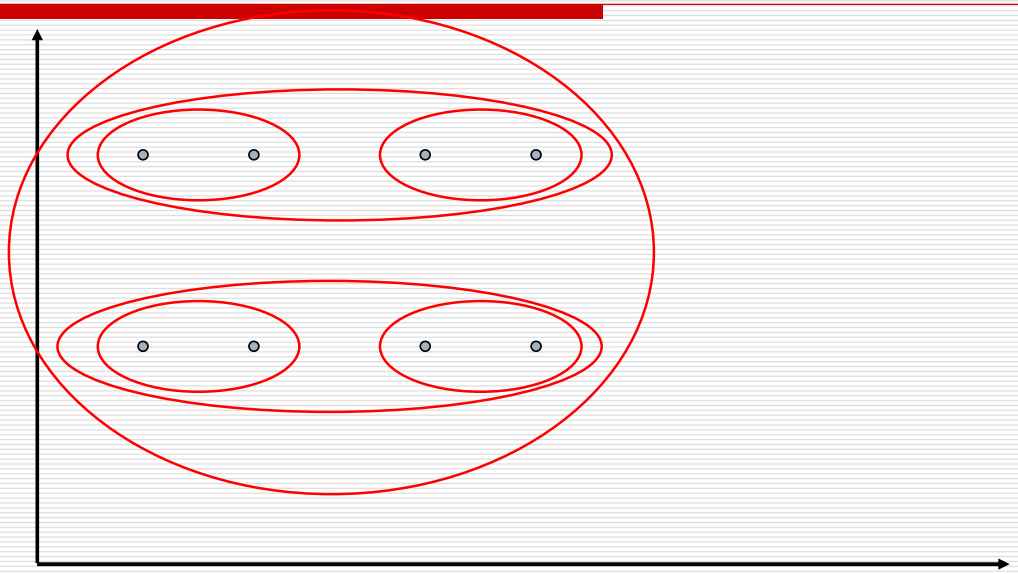
- Use maximum similarity of pairs:

$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

- Can result in "straggly" (long and thin) clusters due to chaining effect.
- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \max(sim(c_i, c_k), sim(c_j, c_k))$$

Single Link Example



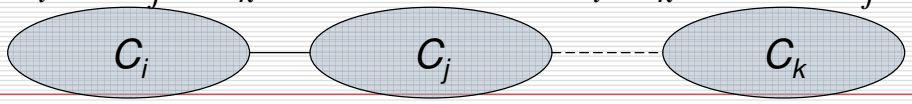
Complete Link Agglomerative Clustering

- Use minimum similarity of pairs:

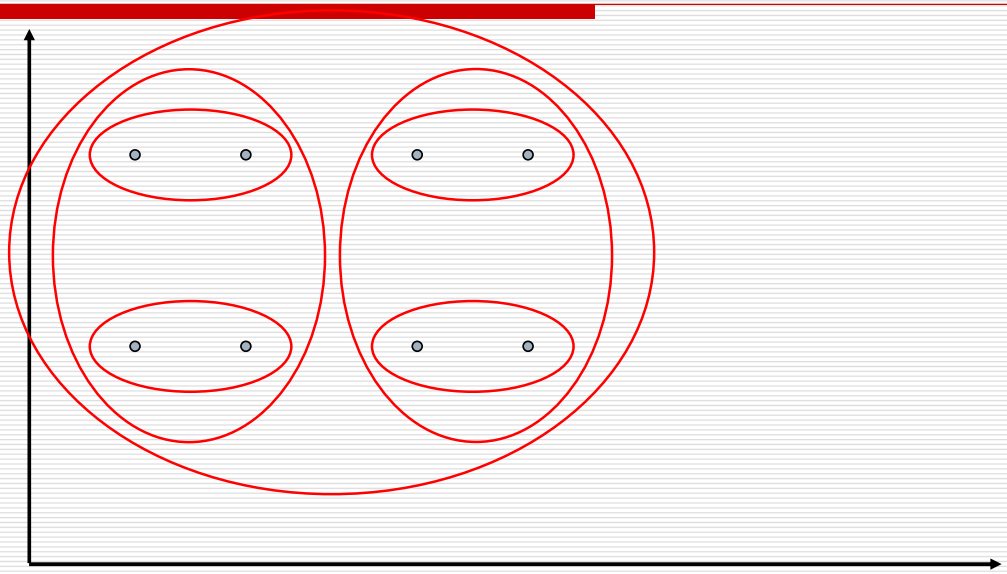
$$sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} sim(x, y)$$

- Makes "tighter," spherical clusters that are typically preferable.
- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \min(sim(c_i, c_k), sim(c_j, c_k))$$



Complete Link Example



Graph theoretical interpretation

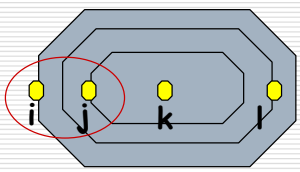
- Single-link Clustering as connected component of a graph
  - If $G(\Delta_k)$ is the graph that links all data points with a distance of at most Δ_k, then the clusters are the connected components of $G(\Delta_k)$
- Complete-link as cliques of a graph
  - If $G(\Delta_k)$ is the graph that links all data points with a distance of at most Δ_k, then the clusters are the cliques of $G(\Delta_k)$
- This motivates the terms single-link and complete-link clustering

Computational Complexity

- In the first iteration, all HAC methods need to compute similarity of all pairs of n individual instances which is $O(n^2)$.
- In each of the subsequent $n-2$ merging iterations, compute the distance between the most recently created cluster and all other existing clusters.
- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.
  - Else $O(n^2 \log n)$ or $O(n^3)$ if done naively

Best-Merge Persistency

- The single-link agglomerative clustering is best-merge persistent
- Suppose that the best merge cluster for k is j
- Then, after merging j with a third cluster $i \neq k$, the merger of i and j will be the k's best merge cluster
- As a consequence, we can keep the best merge candidates for the merged cluster as one of the two best merge candidates for the merged clusters



Single-link and Complete-link drawbacks

- Single link clustering can produce straggling clusters. Since the merge criterion is local, it can cause the chaining effect
- Complete-link clustering pays to much attention to outliers, i.e. points that do not fit well in the global structure of the clusters

Group Average Agglomerative Clustering

- Similarity of two clusters = average similarity of all pairs within merged cluster.

$$sim_{ga}(\omega_i, \omega_j) = \frac{\sum_{d_k \in \omega_i \cup \omega_j} \sum_{d_l \in \omega_i \cup \omega_j, d_l \neq d_k} d_k d_l}{(N_i+N_j)(N_i+N_j-1)}$$

- Compromise between single and complete link.
- An alternative to group-average clustering is centroid clustering

$$sim_{cent}(\omega_i, \omega_j) = \frac{1}{N_i N_j} \sum_{d_k \in \omega_i} \sum_{d_l \in \omega_j, d_l \neq d_k} d_k d_l$$

Computing Group Average and Centroid similarities

- Always maintain sum of vectors in each cluster.

$$s(\omega_j) = \sum_{d_k \in \omega_j} d_k$$

- Compute similarity of clusters in constant time:

$$sim_{ga}(\omega_i, \omega_j) = \frac{s^2(\omega_i \cup \omega_j) - (N_i + N_j)}{(N_i + N_j)(N_i + N_j - 1)}$$

$$sim_{cent}(\omega_i, \omega_j) = \frac{s(\omega_i) s(\omega_j)}{N_i N_j}$$

Summarizing

Single-link	Max sim of any two points	O(N^2)	Chaining effect
Complete-link	Min sim of any two points	O(N^2logN)	Sensitive to outliers
Centroid	Similarity of centroids	O(N^2logN)	Non monotonic
Group-average	Avg sim of any two points	O(N^2logN)	OK