Minimum Spanning Trees

Application to Clustering

Algorithms: Design and Analysis, Part II

Clustering

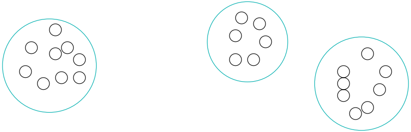
[aka "unsupervised learning"]

Informal goal: Given n "points" [Web pages, images, genome fragments, etc.] classify into "coherent groups".

Assumptions: (1) As input, given a (dis)similarity measure — a distance $d(p, q)$ between each point pair.
(2) Symmetric [i.e., $d(p, q) = d(q, p)$]

Examples: Euclidean distance, genome similarity, etc.

Goal: Same cluster $\iff$ "nearby"



Max-Spacing k-Clusterings

Assume: We know k:= # of clusters desired. [In practice, can experiment with a range of values]

Call points p & q separated if they're assigned to different clusters.

Definition: The spacing of a k-clustering is $\min_{\text{separated } p,q} d(p,q)$. (The bigger the better)

Problem statement: Given a distance measure d and k, compute the k-clustering with maximum spacing.

A Greedy Algorithm

- Initially, each point in a separate cluster
- Repeat until only k clusters:
  - Let $p, q =$ closest pair of separated points (determines the current spacing)
  - Merge the clusters containing p & q into a single cluster.

Note: Just like Kruskal's MST algorithm, but stopped early.
- Points $\leftrightarrow$ vertices, distances $\leftrightarrow$ edge costs, point pairs $\leftrightarrow$ edges.

$\Rightarrow$ Called single-link clustering