Clustering Algorithms: Hierarchical and variations

General agglomerative

• Uses any computable cluster similarity measure \( \text{sim}(C_i, C_j) \)
• For \( n \) objects \( v_1, \ldots, v_n \), assign each to a singleton cluster \( C_i = \{v_i\} \).
• repeat {
  – identify two most similar clusters \( C_j \) and \( C_k \) (could be ties – chose one pair)
  – delete \( C_j \) and \( C_k \) and add \( (C_j \cup C_k) \) to the set of clusters
} until only one cluster
• Dendrograms diagram the sequence of cluster merges.
Agglomerative: remarks

- *Introduction to IR* discusses in great detail for cluster similarity:
  - single-link,
  - complete-link,
  - average of all pairs
  - centroid
- Uses priority queues to get time complexity
  \[ O(n^2 \log n \times (\text{time to compute cluster similarity})) \]
  - one priority queue for each cluster: contains similarities to all other clusters plus bookkeeping info
  - time complexity more precisely:
    \[ O(n^2 \times (\text{time to compute object-object similarity}) + (n^2 \log n \times \text{(time to compute } \text{sim(cluster}_z, \text{ cluster}_j U \text{ cluster}_k) \text{ if know } \text{sim(cluster}_z, \text{ cluster}_j) \text{ and } \text{sim(cluster}_z, \text{ cluster}_k) \}) \]
- Problem with priority queue?

Single pass agglomerative-like

Given arbitrary order of objects to cluster: \( v_1, \ldots, v_n \)
and threshold \( \tau \)
Put \( v_1 \) in cluster \( C_1 \) by itself
For \( i = 2 \) to \( n \) {
  for all existing clusters \( C_j \)
    calculate \( \text{sim}(v_i, C_j) \);
  record most similar cluster to \( v_i \) as \( C_{\text{max}(i)} \)
  if \( \text{sim}(v_i, C_{\text{max}(i)}) > \tau \) add \( v_i \) to \( C_{\text{max}(i)} \)
  else create new cluster \{\( v_i \}\)
}
Issues

• put $v_i$ in cluster after seeing only $v_1, \ldots v_{i-1}$
• not hierarchical
• tends to produce large clusters
  – depends on $\tau$
• depends on order of $v_i$
Alternate perspective for single-link algorithm

• Build a **minimum spanning tree (MST)** - graph alg.
  – edge weights are pair-wise similarities
  – since in terms of similarities, not distances, really want maximum spanning tree
• For some threshold $\tau$, remove all edges of similarity $< \tau$
• Tree falls into pieces $\Rightarrow$ clusters

• Not hierarchical, but get hierarchy for sequence of $\tau$