10d Machine Learning: Symbol-based

More clustering examples

Additional references for the slides:
David Grossman's clustering slides:
http://ir.iit.edu/~dagr/IRcourse/Notes/08Clustering.pdf
Subbarao Kambhampati's clustering slides:
http://rakaposhi.eas.asu.edu/cse494/notes/f02-clustering.ppt

Document clustering

Automatically group related documents into clusters given some measure of similarity. For example,

- **medical documents**
- **legal documents**
- **financial documents**
- **web search results**

Hierarchical Agglomerative Clustering (HAC)

- **Given n documents, create a n x n doc-doc similarity matrix.**
- **Each document starts as a cluster of size one.**
- **do until there is only one cluster**
  - **Combine the two clusters with the greatest similarity (if X and Y are the most mergable pair of clusters, then we create X-Y as the parent of X and Y. Hence the name “hierarchical”.)**
  - **Update the doc-doc matrix.**

Example

Consider A, B, C, D, E as documents with the following similarities:

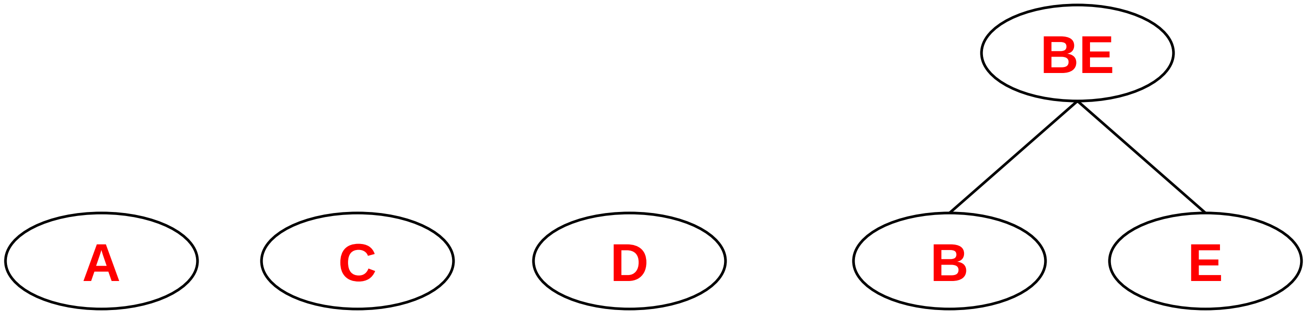
	A	B	C	D	E
A	-	2	7	9	4
B	2	-	9	11	14
C	7	9	-	4	8
D	9	11	4	-	2
E	4	14	8	2	-

The pair with the highest similarity is:

B-E = 14

Example

So let's cluster B and E. We now have the following structure:



Example

Update the doc-doc matrix:

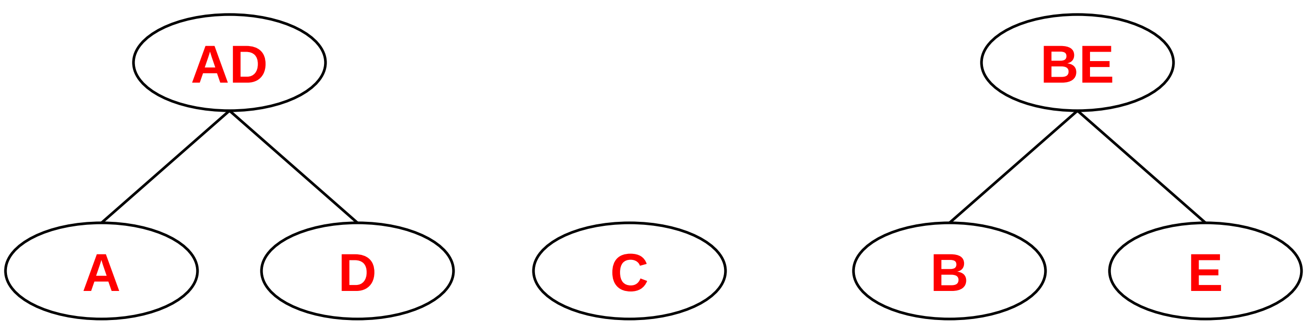
	A	BE	C	D
A	-	2	7	9
BE	2	-	8	2
C	7	8	-	4
D	9	2	4	-

To compute (A,BE): take the minimum of (A,B)=2 and (A,E)=4.

This is called *complete linkage*.

Example

Highest link is A-D. So let's cluster A and D. We now have the following structure:



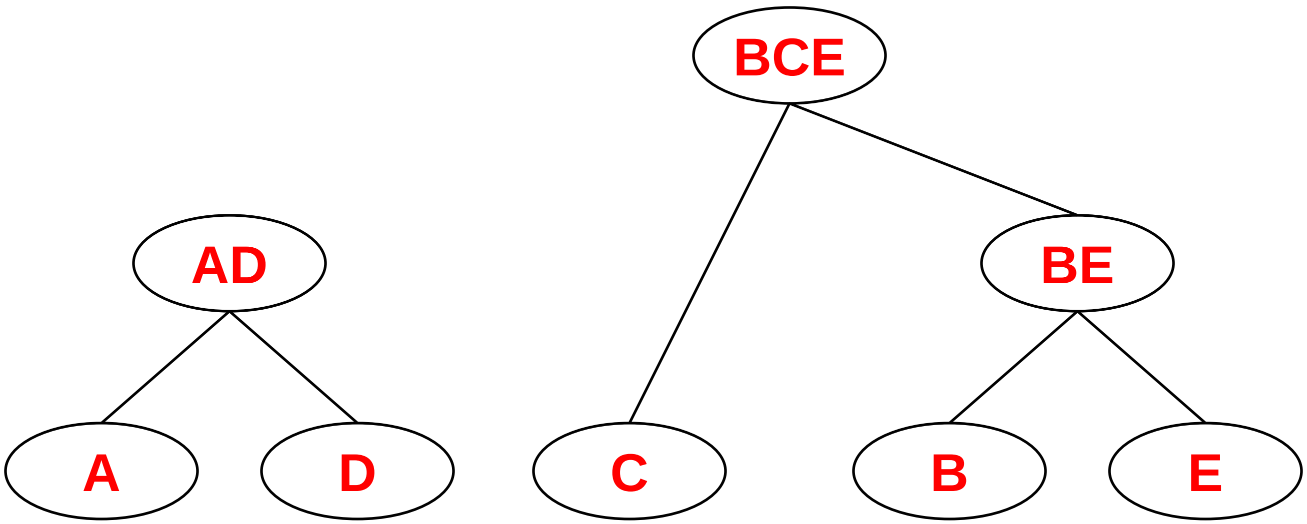
Example

Update the doc-doc matrix:

	AD	BE	C
AD	-	2	4
BE	2	-	8
C	4	8	-

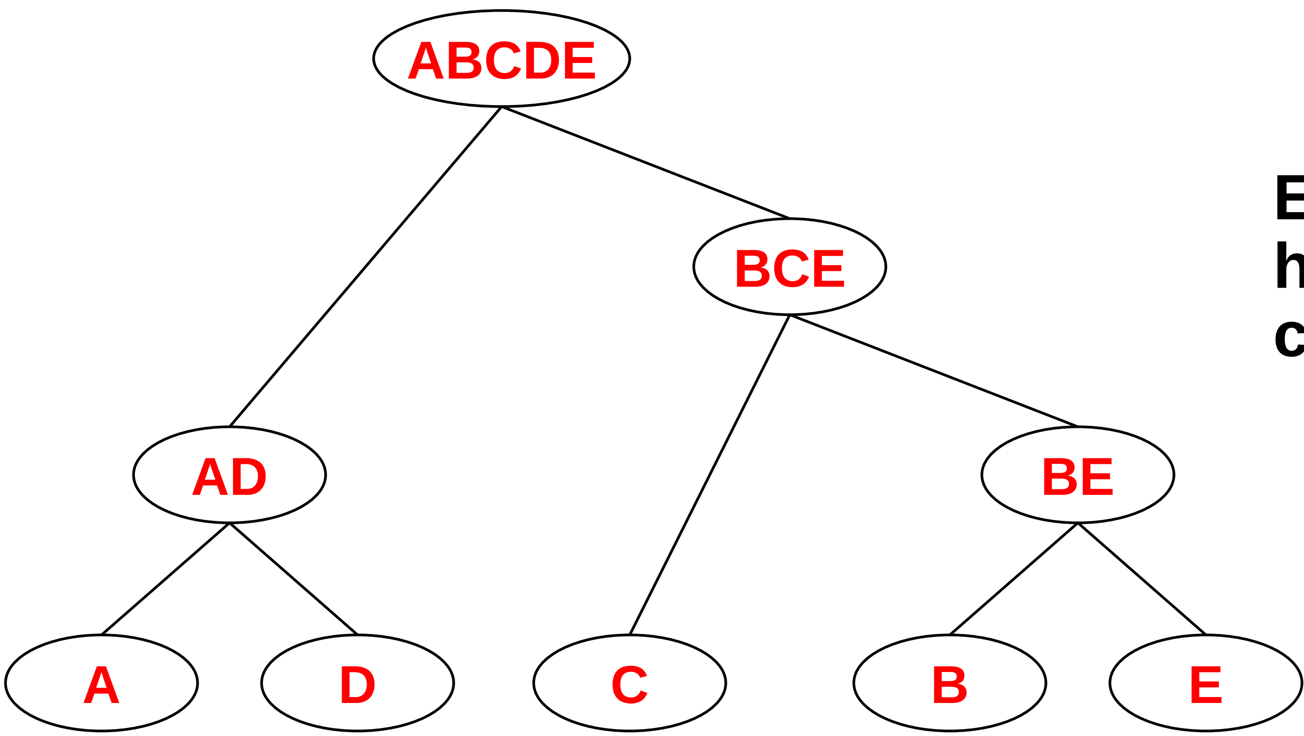
Example

- **Highest link is BE-C. So let's cluster BE and C. We now have the following structure:**



Example

- At this point, there are only two nodes that have not been clustered. So we cluster AD and BCE. We now have the following structure:



Everything has been clustered.

Time complexity analysis

Hierarchical agglomerative clustering (HAC) requires:

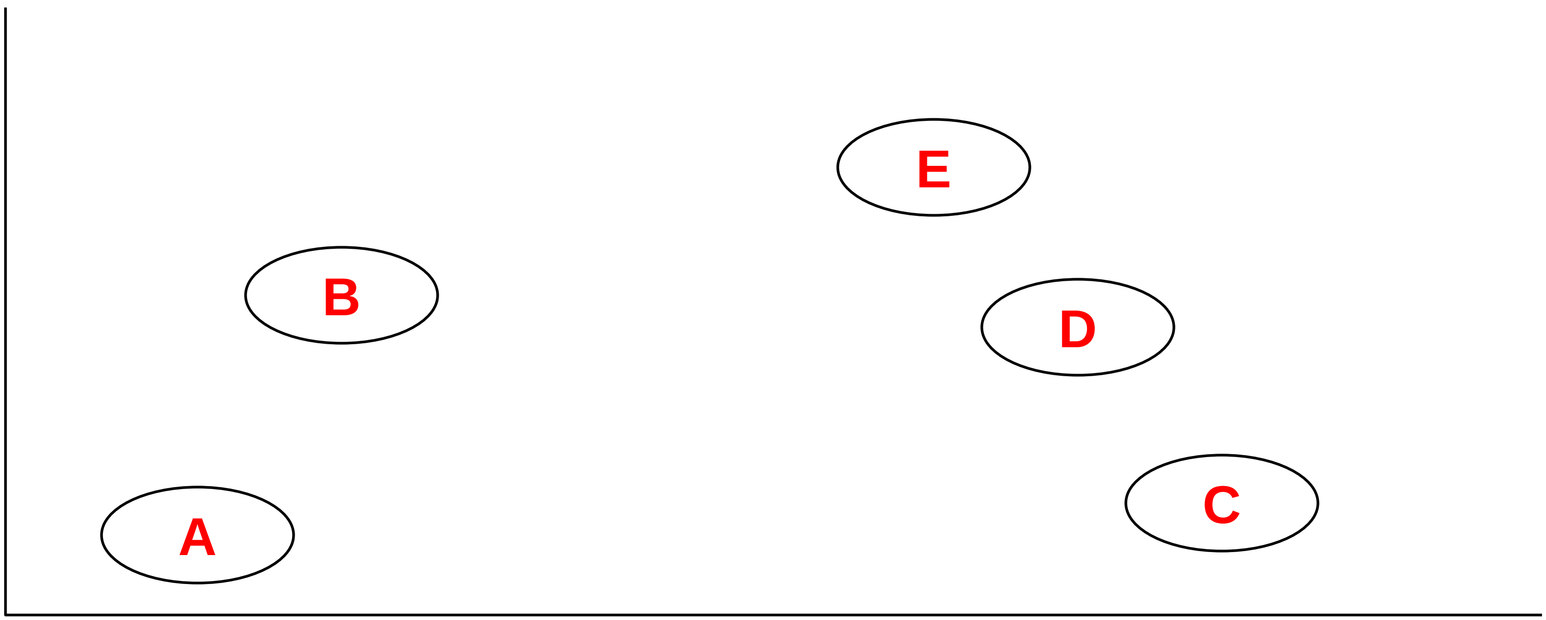
- **$O(n^2)$ to compute the doc-doc similarity matrix**
- **One node is added during each round of clustering so there are now O(n) clustering steps**
- **For each clustering step we must re-compute the doc-doc matrix. This requires O(n) time.**
- **So we have: $n^2 + (n)(n) = O(n^2)$ – so it's expensive!**
- **For 500,000 documents n^2 is 250,000,000,000!!**

One pass clustering

- **Choose a document and declare it to be in a cluster of size 1.**
- **Now compute the distance from this cluster to all the remaining nodes.**
- **Add “closest” node to the cluster. If no node is really close (within some threshold), start a new cluster between the two closest nodes.**

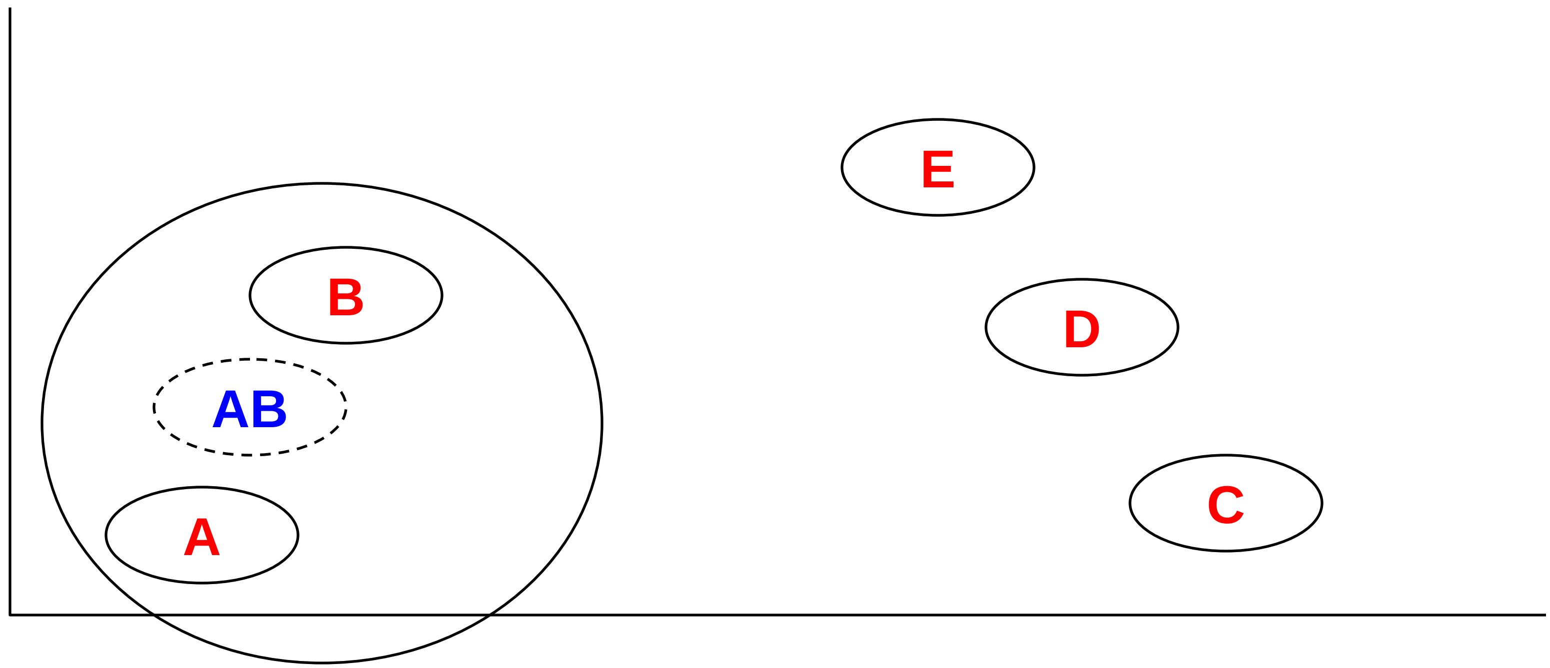
Example

- Consider the following nodes



Example

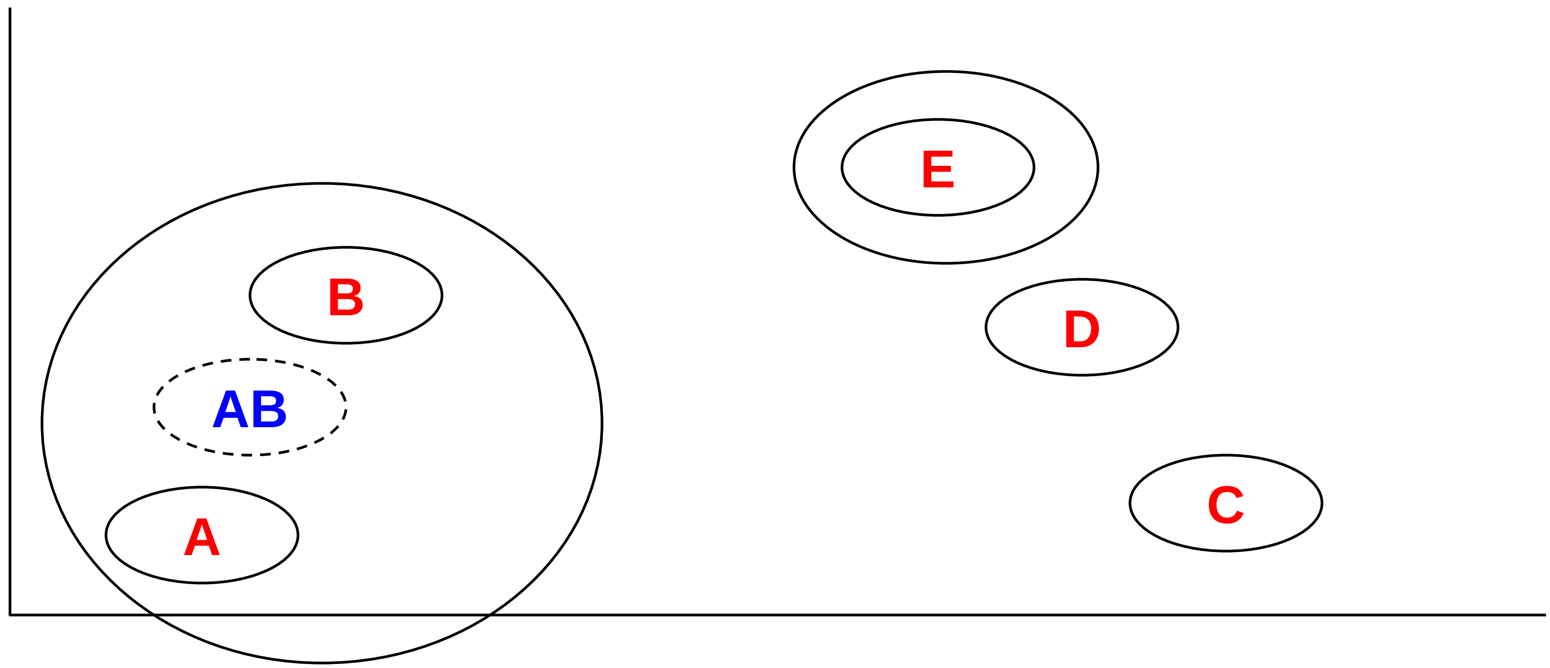
- Choose node A as the first cluster
- Now compute the distance between A and the others. B is the closest, so cluster A and B.
- Compute the centroid of the cluster just formed.



Example

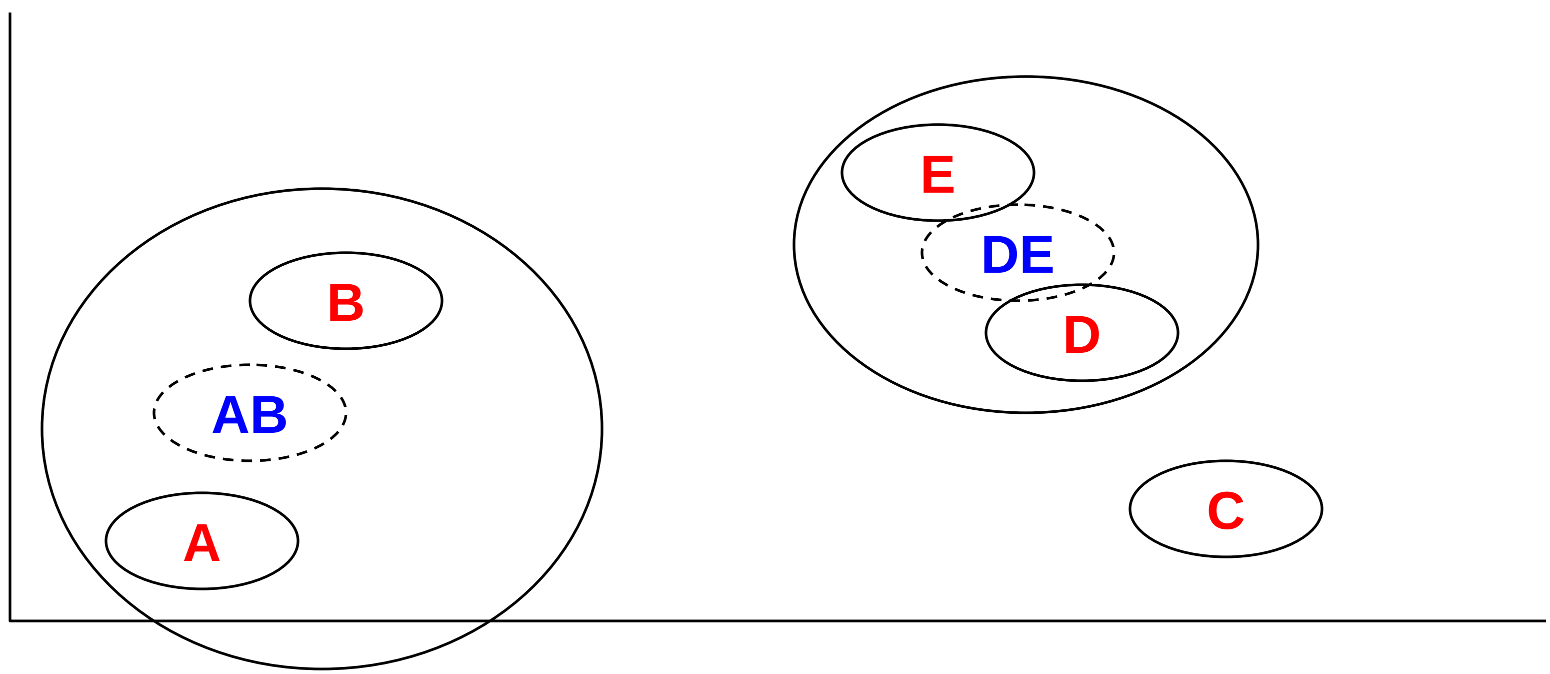
• **Compute the distance between A-B and all the remaining clusters using the centroid of A-B.**

• **Let's assume all the others are too far from AB. Choose one of these non-clustered elements and place it in a cluster. Let's choose E.**



Example

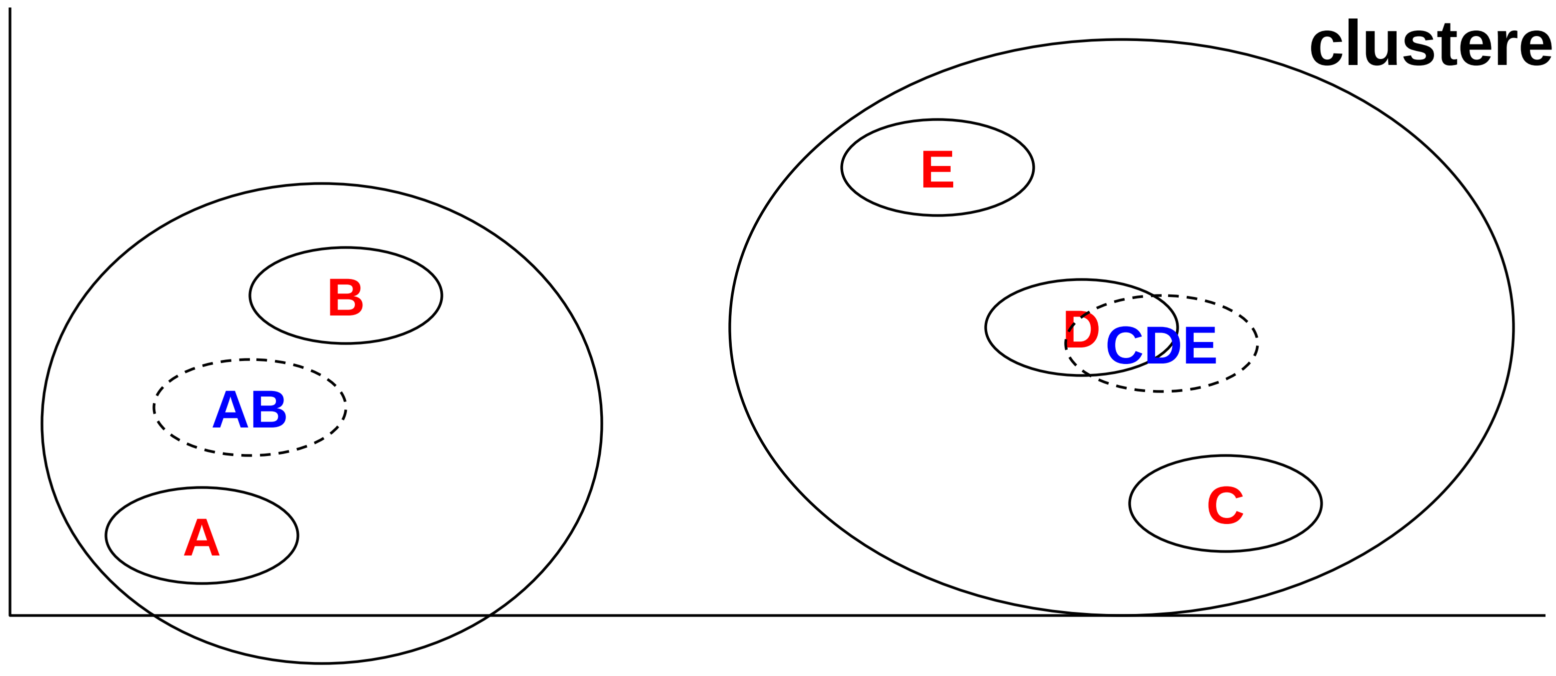
- Compute the distance from E to D and E to C.
- E to D is closer so we form a cluster of E and D.



Example

- Compute the distance from D-E to C.
- It is within the threshold so include C in this cluster.

Everything has been clustered.



Time complexity analysis

One pass requires:

- n passes as we add node for each pass
- First pass requires n-1 comparisons
- Second pass requires n-2 comparisons
- Last pass needs 1
- So we have $1 + 2 + 3 + \ldots + (n-1) = (n-1)(n) / 2$
- $(n^2 - n) / 2 = O(n^2)$
- The constant is lower for one pass but we are still at n^2 .

Remember k-means clustering

- Pick k points as the seeds of k clusters
- At the onset, there are k clusters of size one.
- do until all nodes are clustered
  - Pick a point and put it into the cluster whose centroid is closest.
  - Recompute the centroid of the modified cluster.

Time complexity analysis

K-means requires:

• Each node gets added to a cluster, so there are n clustering steps

• For each addition, we need to compare to k centroids

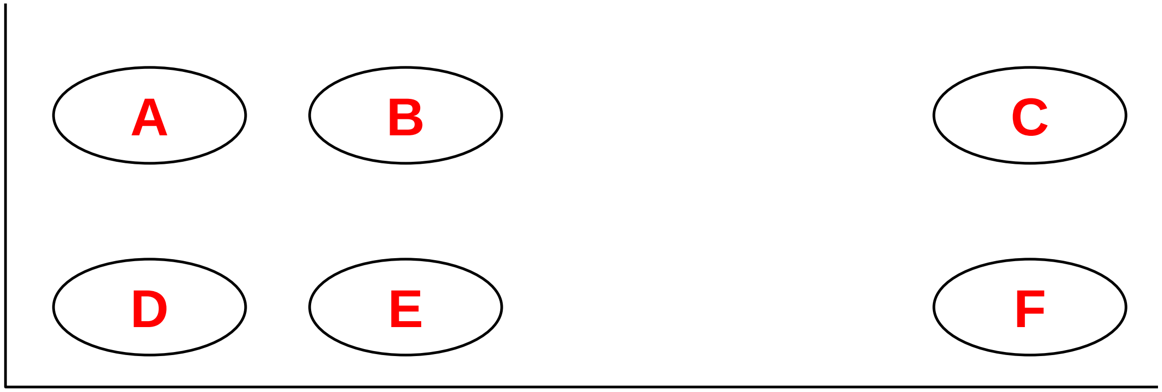
• We also need to recompute the centroid after adding the new node, this takes a constant amount of time (say c)

• The total time needed is (k + c) n = O(n)

• So it is a linear algorithm!

But there are problems…

- K needs to be known in advance or need trials to compute k
- Tends to go to local minima that are sensitive to the starting centroids:



If the seeds are B and E, the resulting clusters are {A,B,C} and {D,E,F}.

If the seeds are D and F, the resulting clusters are {A,B,D,E} and {C,F}.

Two questions for you

1. Why did the computer go to the restaurant?

2. What do you do when you have a slow algorithm that produces quality results, and a fast algorithm that cannot guarantee quality?

1. To get a byte.

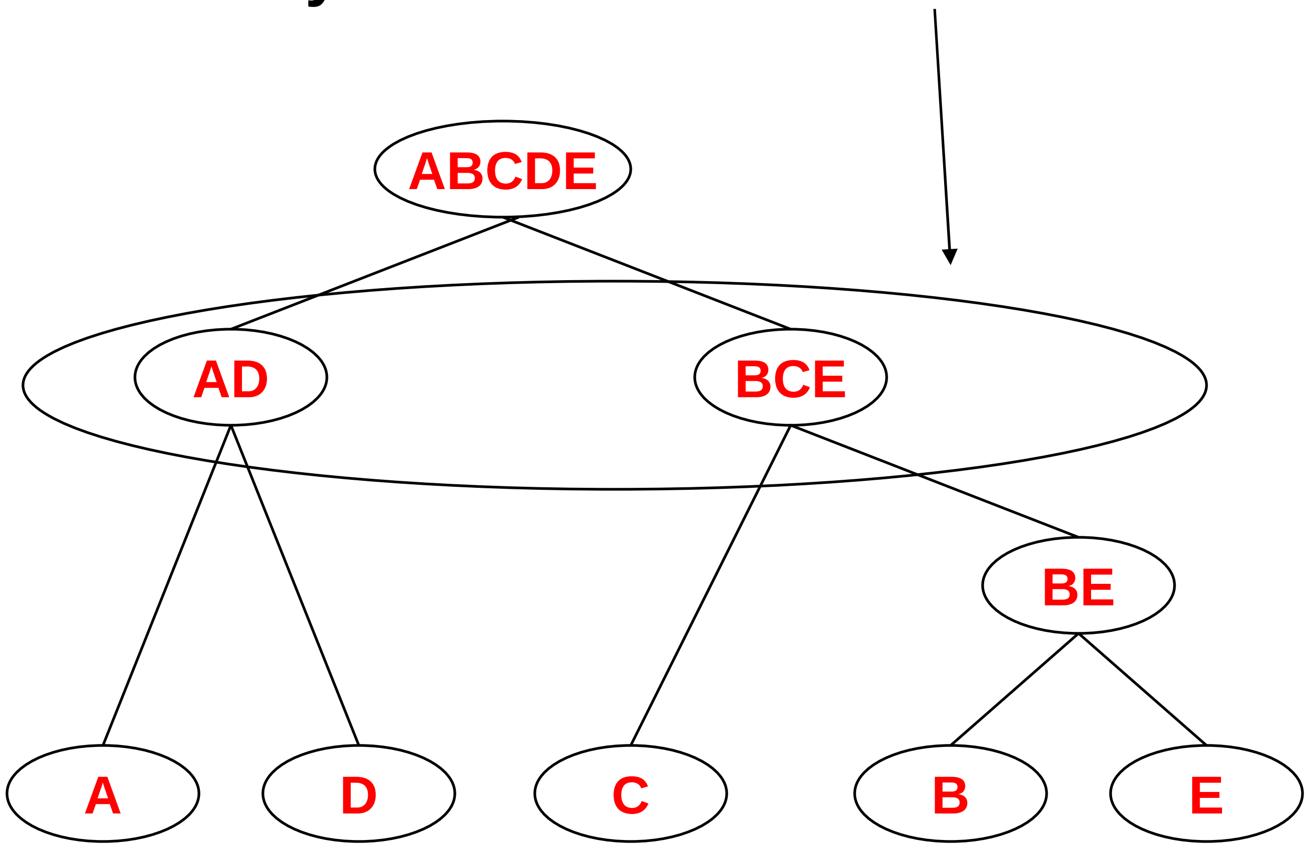
2. Many things…
   One option is to use the slow algorithm on a portion of the problem to obtain a better starting point for the fast algorithm.

Buckshot clustering

- The goal is to reduce the run time by combining HAC and k-means clustering.
- Select d documents where d is SQRT(n).
- Cluster these d documents using HAC, this will take O(n) time.
- Use the results of HAC as initial seeds for k-means.
- It uses HAC to bootstrap k-means.
- The overall algorithm is O(n) and avoids problems of bad seed selection.

Getting the k clusters

Cut where you have k clusters



Effect of document order

- With hierarchical clustering we get the same clusters every time.
- With one pass clustering, we get different clusters based on the order we process the documents.
- With k-means clustering, we get different clusters based on the selected seeds.

Computing the distance (time)

- In our time complexity analysis we finessed the time required to compute the distance between two nodes
- Sometimes this is an expensive task depending on the analysis required

Computing the distance (methods)

- **To compute the intra-cluster distance: (Sum/min/max/avg) the (absolute/squared) distance between**
  - **All pairs of points in the cluster, or**
  - **Between the centroid and all points in the cluster**
- **To compute the inter-cluster distance for HAC:**
  - **Single-link: distance between closest neighbors**
  - **Complete-link: distance between farthest neighbors**
  - **Group-average: average distance between all pairs of neighbors**
  - **Centroid-distance: distance between centroids (most commonly used)**

Measuring the quality of the clusters

A good clustering is one where

• (intra-cluster distance) the sum of distances between objects in the same cluster are minimized

• (inter-cluster distance) while the distances between different clusters are maximized

The objective is to minimize: F(intra, inter)

How many possible clusterings?

If we have n points and would like to cluster them into k clusters, then there are k clusters the first point can go to, there are k clusters for each of the remaining points. So the total number of possible clusterings is k^n.

Brute force enumeration will not work. That is why we have iterative optimization algorithms that start with a clustering and iteratively improve it.

Finally, note that noise (outliers) is a problem for clustering too. One can use statistical techniques to identify outliers.