ALDA Fall 2021 – Homework 5

GITHUB Repository - https://github.ncsu.edu/sjanard/engr-ALDA-fall2021-H12

Homework Team 12 – HW12

Sudharsan Janardhanan – sjanard
Sriram Sudharsan – ssudhar
Pradyumna Vemuri – pvemuri

1. (10 points) [K-means Clustering] [John Wesley Hostetter (Designed) & Chengyuan Liu (Graded)] Using K-means clustering and Euclidean distance, cluster the 11 data points in Figure 1 into three clusters. We assume that the initial seeds are at points E, F, and J (in yellow). Answer the following questions:

(a) (4 points) Run the K-means algorithm for one round. Calculate the coordinates of the new centroids. What are the new clusters? Show your work in the first subgraph in Figure 2.

Find the complete version of the code snippets attached to the document in the file HW5_1+2.py in the GitHub repository engr-ALDA-fall2021-H12

ROUND 1:

Centroids: E=(-8,0), F=(-2,-1), J=(0,-2)

Euclidean distance between each centroid to a given point

R1	C1	C2	C3
A	3.162	7.2801	9.055
B	2.236	4.472	6.708
C	4.2426	5	7.07
D	8.062	2	1
E	0	6.082	8.24
F	6.08	0	2.236
G	4.47	3.605	5.65
H	2.236	8	10.04
I	4.47	2.236	4
J	8.24	2.236	0
K	4.12	5.83	7.28

CLUSTERS AFTER ROUND 1

C1 = (A,B,C,E,H,K)
C2: (F,G,I)
C3: (D,J)

ROUND 2:

Centroids: C1 = (-7.5,-0.666), C2 = (-3.333,-0.333), C3 = (0, -1.5)

Euclidean distance between each centroid to a given point

R2	C1	C2	C3
A	2.773	6.262	9.124
B	2.242	2.981	6.5
C	4.437	3.726	6.7268
D	7.507	3.399	0.5
E	0.833	4.678	8.139
F	5.51	1.49	2.061
G	4.4	2.426	5.315
H	2.522	6.699	10.012
I	3.745	1.795	4.031
J	7.617	3.726	0.5
K	3.37	5.185	7.433

CLUSTERS AFTER ROUND 2

C1: (A,B,E,H,K)

C2: (C,F,G,I)

C3: (D,J)

ROUND 3:

Centroids: C1 = (-8,-1.4). C2 = (-3.75,0.5), C3 = (0, -1.5)

Euclidean distance between each centroid to a given point

R3	C1	C2	C3
A	1.886	6.309	9.214
B	3.124	2.304	6.5
C	5.325	2.795	6.702
D	8.009	4.038	0.5
E	1.4	4.279	8.139
F	6.013	2.304	2.061
G	5.24	1.52	5.315
H	2.039	6.427	10.012
I	4.044	2.512	4.031
J	8.022	4.506	0.5
K	2.785	5.55	7.433

CLUSTERS AFTER ROUND 3

C1: (A,E,H,K)

C2: (B,C,G,I)

C3: (D,F,J)

ROUND 4:

Centroids: C1 = (-8.5,-2), C2 = (-4.75,1), C3 = (-0.666, -1.333)

Euclidean distance between each centroid to a given point

R4	C1	C2	C3
A	1.118	5.836	8.499
B	3.905	1.25	5.821
C	6.103	2.015	6.128
D	8.558	5.153	0.744
E	2.061	3.4	7.454
F	6.576	3.4	1.374
G	6.02	1.25	4.714
H	1.802	5.618	9.339
I	4.5	3.092	3.4
J	8.5	5.618	0.942
K	2.5	5.482	6.872

CLUSTERS AFTER ROUND 4:

C1: (A,E,H,K)

C2: (B,C,G,I)

C3: (D,F,J)

ROUND 5:

Centroids: C1 = (-8.5,-2), C2 = (-4.75,1), C3 = (-0.666, -1.333)

Euclidean distance between each centroid to a given point

R5	C1	C2	C3
A	1.118	5.836	8.499
B	3.905	1.25	5.821
C	6.103	2.015	6.128
D	8.558	5.153	0.744
E	2.061	3.4	7.454
F	6.576	3.4	1.374
G	6.02	1.25	4.714
H	1.802	5.618	9.339

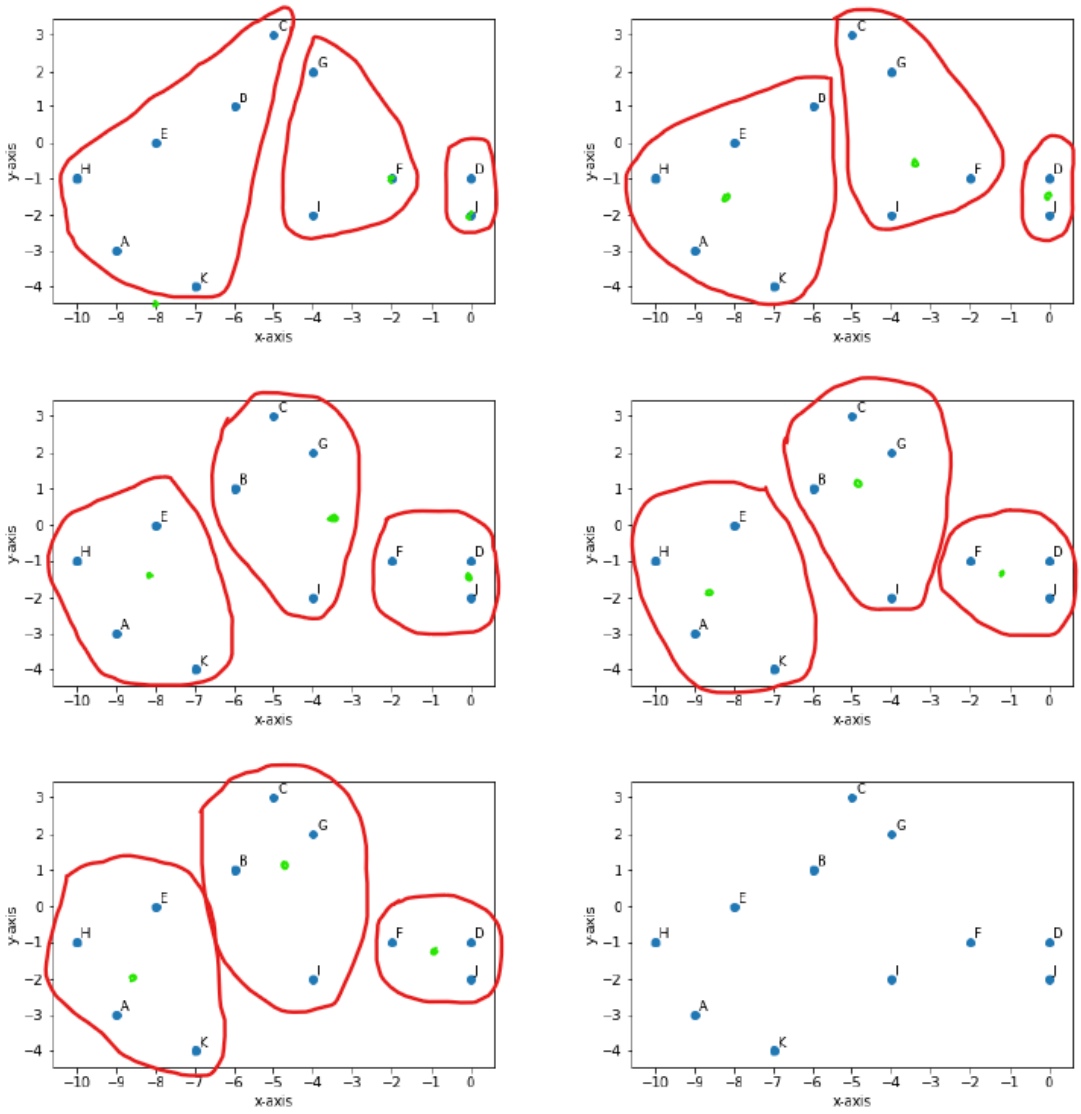
CLUSTERS AFTER ROUND 5:

C1: (A,E,H,K)

C2: (B,C,G,I)

C3: (D,F,J)

(b) (6 points) How many rounds are needed for the K-means clustering algorithm to converge? Draw the resulting clusters and new centroid at the end of each round (including the first round) in the Figure 2. Indicate the coordinates alongside corresponding centroids. Add new graphics if needed; Stop when the algorithm converges and clearly label on the graph where the algorithm converges.



ROUND 1:

Centroids: E = (-8,0), F = (-2,-1), J = (0,-2)

ROUND 2:

Centroids: C1 = (-7.5,-0.666), C2 = (-3.333,-0.333), C3 = (0, -1.5)

ROUND 3:
Centroids: C1 = (-8,-1.4). C2 = (-3.75,0.5), C3 = (0, -1.5)

ROUND 4:
Centroids: C1 = (-8.5,-2), C2 = (-4.75,1), C3 = (-0.666, -1.333)

ROUND 5:
Centroids: C1 = (-8.5,-2), C2 = (-4.75,1), C3 = (-0.666, -1.333)

2. (15 points) [Hierarchical Clustering] [John Wesley Hostetter(Designed) & Chengyuan Liu (Graded)] We will use the same dataset as in Question 1 for the following problem. The Euclidean Distance matrix between each pair of the data points is listed in the figure below:

(a) (8 points) Perform single and complete link hierarchical clustering. Show your results by drawing corresponding dendrogram. The dendrogram should clearly show the order and the height in which the clusters are merged. In case of a tie please resolve in alphabetical order of the points' labels. NO PARTIAL CREDIT.

LIST OF COORDINATES

	x	y
A	-9	-3
B	-6	1
C	-5	3
D	0	-1
E	-8	0
F	-2	-1
G	-4	2
H	-10	-1
I	-4	-2
J	0	-2
K	-7	-4

```
from sklearn.metrics.pairwise import euclidean_distances
mat = euclidean_distances(df)
euc_matrix = pd.DataFrame(mat, columns = df.index)
euc_matrix.to_csv('euc_mat.csv')
```

SINGLE LINK HIERARCHICAL CLUSTERING

- Single link hierarchical clustering is performed by considering the smallest distance in the distance matrix and clustering both the points, say A, B associated with that distance.

- We recalculate the distance from other points to the two points in consideration by taking the minimum distance to either one of the points. For E.g. If point C has distance 4 to A and distance 3 to B, we ignore the larger of the two distances and keep the smaller.

- We perform this operation until we are left with a single cluster

Note: The Euclidean distance marked in red gives us the smallest distance and the points we merge.

ROUND 1:

	A	B	C	D	E	F	G	H	I	J	K
A	0	5	7.211103	9.219544	3.162278	7.28011	7.071068	2.236068	5.09902	9.055385	2.236068
B	5	0	2.236068	6.324555	2.236068	4.472136	2.236068	4.472136	3.605551	6.708204	5.09902
C	7.211103	2.236068	0	6.403124	4.242641	5	1.414214	6.403124	5.09902	7.071068	7.28011
D	9.219544	6.324555	6.403124	0	8.062258	2	5	10	4.123106	1	7.615773
E	3.162278	2.236068	4.242641	8.062258	0	6.082763	4.472136	2.236068	4.472136	8.246211	4.123106
F	7.28011	4.472136	5	2	6.082763	0	3.605551	8	2.236068	2.236068	5.830952
G	7.071068	2.236068	1.414214	5	4.472136	3.605551	0	6.708204	4	5.656854	6.708204
H	2.236068	4.472136	6.403124	10	2.236068	8	6.708204	0	6.082763	10.04988	4.242641
I	5.09902	3.605551	5.09902	4.123106	4.472136	2.236068	4	6.082763	0	4	3.605551
J	9.055385	6.708204	7.071068	1	8.246211	2.236068	5.656854	10.04988	4	0	7.28011
K	2.236068	5.09902	7.28011	7.615773	4.123106	5.830952	6.708204	4.242641	3.605551	7.28011	0

ROUND 2:

	A	B	C	D,J	E	F	G	H	I	K
A	0									
B	5	0								
C	7.211103	2.236068	0							
D,J	9.219544	6.324555	6.403124	0						
E	3.162278	2.236068	4.242641	8.062258						
F	7.28011	4.472136	5	2	6.082763	0				
G	7.071068	2.236068	1.414214	5	4.472136	3.605551	0			
H	2.236068	4.472136	6.403124	10	2.236068	8	6.708204	0		
I	5.09902	3.605551	5.09902	4	4.472136	2.236068	4	6.082763	0	
K	2.236068	5.09902	7.28011	7.28011	4.123106	5.830952	6.708204	4.242641	3.605551	0

ROUND 3:

	A	B	C,G	D,J	E	F	H	I	K
A	0								
B	5	0							
C,G	7.071068	2.236068	0						
D,J	9.055385	6.324555	5	0					
E	3.162278	2.236068	4.242641	8.062258	0				
F	7.28011	4.472136	3.605551	2	6.082763	0			
H	2.236068	4.472136	6.403124	10	2.236068	8	0		
I	5.09902	3.605551	4	4	4.472136	2.236068	6.082763	0	
K	2.236068	5.09902	6.708204	7.28011	4.123106	5.830952	4.242641	3.605551	0

ROUND 4:

	A	B	C,G	D,F,J	E	H	I	K
A	0							
B	5	0						
C,G	7.071068	2.236068	0					
D,F,J	7.28011	4.472136	3.605551	0				
E	3.162278	2.236068	4.242641	6.082763	0			
H	2.236068	4.472136	6.403124	8	2.236068	0		
I	5.09902	3.605551	4	2.236068	4.472136	6.082763	0	
K	2.236068	5.09902	6.708204	5.830952	4.123106	4.242641	3.605551	0

ROUND 5:

	A,H	B	C,G	D,F,J	E	I	K
A,H	0						
B	4.472136	0					
C,G	6.403124	2.236068	0				
D,F,J	7.28011	4.472136	3.605551	0			
E	2.236068	2.236068	4.242641	6.082763	0		
I	5.09902	3.605551	4	2.236068	4.472136	0	
K	2.236068	5.09902	6.708204	5.830952	4.123106	3.605551	0

ROUND 6:

	A,E,H	B	C,G	D,F,J	I	K
A,E,H	0					
B	2.236068	0				
C,G	4.242641	2.236068	0			
D,F,J	6.082763	4.472136	3.605551	0		
I	4.472136	3.605551	4	2.236068	0	
K	2.236068	5.09902	6.708204	5.830952	3.605551	0

ROUND 7:

	A,B,E,H	C,G	D,F,J	I	K
A,B,E,H	0				
C,G	2.236068	0			
D,F,J	4.472136	3.605551	0		
I	3.605551	4	2.236068	0	
K	2.236068	6.708204	5.830952	3.605551	0

ROUND 8:

	A,B,C,E,G,H	D,F,J	I	K
A,B,C,E,G,H	0			
D,F,J	3.605551275	0		
I	3.605551275	2.236068	0	
K	2.236067977	5.830952	3.605551	0

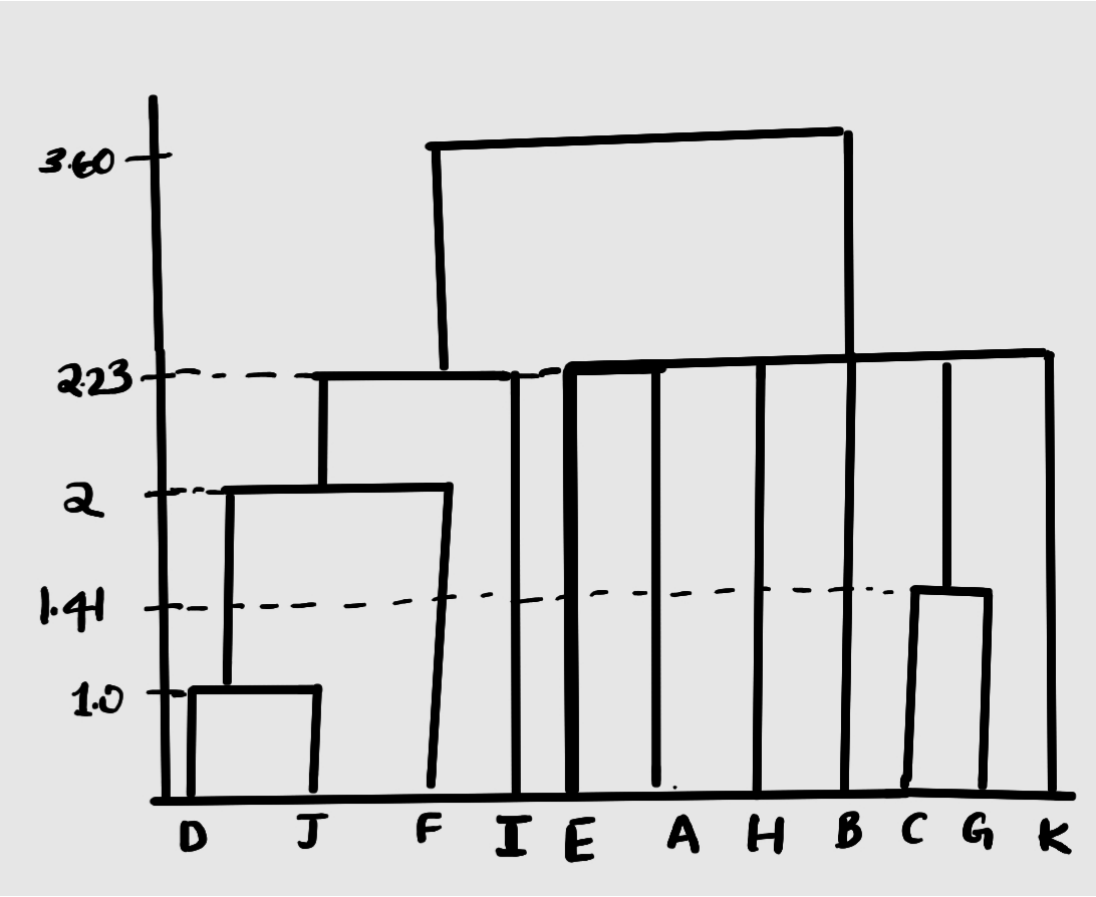
ROUND 9:

	A,B,C,E,G,H,K	D,F,J	I
A,B,C,E,G,H, K	0		
D,F,J	3.605551275	0	
I	3.605551275	2.236068	0

ROUND 10:

	A,B,C,E,G,H,K	D,F,I,J
A,B,C,E,G,H,K	0	
D,F,I,J	3.605551275	0

DENDOGRAM FOR SINGLE LINK MATRIX



COMPLETE LINK HIERARCHICAL CLUSTERING

- Complete link hierarchical clustering is performed by considering the smallest distance in the distance matrix and clustering both the points, say A, B associated with that distance.

- We recalculate the distance from other points to the two points in consideration by taking the maximum distance to either one of the points. For E.g. If point C has distance 4 to A and distance 3 to B, we ignore the smaller of the two distances and keep the larger one.

- We perform this operation until we are left with a single cluster

Note: The Euclidean distance marked in red gives us the smallest distance and the points we merge.

ROUND 1:

	A	B	C	D	E	F	G	H	I	J	K
A	0	5	7.211103	9.219544	3.162278	7.28011	7.071068	2.236068	5.09902	9.055385	2.236068
B	5	0	2.236068	6.324555	2.236068	4.472136	2.236068	4.472136	3.605551	6.708204	5.09902
C	7.211102551	2.236068	0	6.403124	4.242641	5	1.414214	6.403124	5.09902	7.071068	7.28011
D	9.219544457	6.324555	6.403124	0	8.062258	2	5	10	4.123106	1	7.615773
E	3.16227766	2.236068	4.242641	8.062258	0	6.082763	4.472136	2.236068	4.472136	8.246211	4.123106
F	7.280109889	4.472136	5	2	6.082763	0	3.605551	8	2.236068	2.236068	5.830952
G	7.071067812	2.236068	1.414214	5	4.472136	3.605551	0	6.708204	4	5.656854	6.708204
H	2.236067977	4.472136	6.403124	10	2.236068	8	6.708204	0	6.082763	10.04988	4.242641
I	5.099019514	3.605551	5.09902	4.123106	4.472136	2.236068	4	6.082763	0	4	3.605551
J	9.055385138	6.708204	7.071068	1	8.246211	2.236068	5.656854	10.04988	4	0	7.28011
K	2.236067977	5.09902	7.28011	7.615773	4.123106	5.830952	6.708204	4.242641	3.605551	7.28011	0

ROUND 2:

	A	B	C	D,J	E	F	G	H	I	K
A	0									
B	5	0								
C	7.211102551	2.236068	0							
D,J	9.219544457	6.708204	7.071068	0						
E	3.16227766	2.236068	4.242641	8.246211						
F	7.280109889	4.472136	5	2.236068	6.082763	0				
G	7.071067812	2.236068	1.414214	5.656854	4.472136	3.605551	0			
H	2.236067977	4.472136	6.403124	10.04988	2.236068	8	6.708204	0		
I	5.099019514	3.605551	5.09902	4.123106	4.472136	2.236068	4	6.082763	0	
K	2.236067977	5.09902	7.28011	7.615773	4.123106	5.830952	6.708204	4.242641	3.605551	0

ROUND 3:

	A	B	C,G	D,J	E	F	H	I	K
A	0								
B	5	0							
C,G	7.211102551	2.236068	0						
D,J	9.219544457	6.708204	7.071068	0					
E	3.16227766	2.236068	4.472136	8.246211	0				
F	7.280109889	4.472136	5	2.236068	6.082763	0			
H	2.236067977	4.472136	6.708204	10.04988	2.236068	8	0		
I	5.099019514	3.605551	5.09902	4.123106	4.472136	2.236068	6.082763	0	
K	2.236067977	5.09902	7.28011	7.615773	4.123106	5.830952	4.242641	3.605551	0

ROUND 4:

	A,H	B	C,G	D,J	E	F	I	K
A,H	0							
B	5	0						
C,G	7.211102551	2.236068	0					
D,J	10.04987562	6.708204	7.071068	0				
E	3.16227766	2.236068	4.472136	8.246211	0			
F	7.280109889	4.472136	5	2.236068	6.082763	0		
I	6.08276253	3.605551	5.09902	4.123106	4.472136	2.236068	0	
K	4.242640687	5.09902	7.28011	7.615773	4.123106	5.830952	3.605551	0

ROUND 5:

	A,H	B,C,G	D,J	E	F	I	K
A,H	0						
B,C,G	7.211102551	0					
D,J	10.04987562	7.071068	0				
E	3.16227766	4.472136	8.246211	0			
F	7.280109889	5	2.236068	6.082763	0		
I	6.08276253	5.09902	4.123106	4.472136	2.236068	0	
K	4.242640687	7.28011	7.615773	4.123106	5.830952	3.605551	0

ROUND 6:

	A,H	B,C,G	D,F,J	E	I	K
A,H	0					
B,C,G	7.211102551	0				
D,F,J	10.04987562	7.071068	0			
E	3.16227766	4.472136	8.246211	0		
I	6.08276253	5.09902	4.123106	4.472136	0	
K	4.242640687	7.28011	7.615773	4.123106	3.605551	0

ROUND 7:

	A,E,H	B,C,G	D,F,J	I	K
A,E,H	0				
B,C,G	7.211102551	0			
D,F,J	10.04987562	7.071068	0		
I	6.08276253	5.09902	4.123106	0	
K	4.242640687	7.28011	7.615773	3.605551	0

ROUND 8:

	A,E,H	B,C,G	D,F,J	I,K
A,E,H	0			
B,C,G	7.211102551	0		
D,F,J	10.04987562	7.071068	0	
I,K	6.08276253	7.28011	7.615773	0

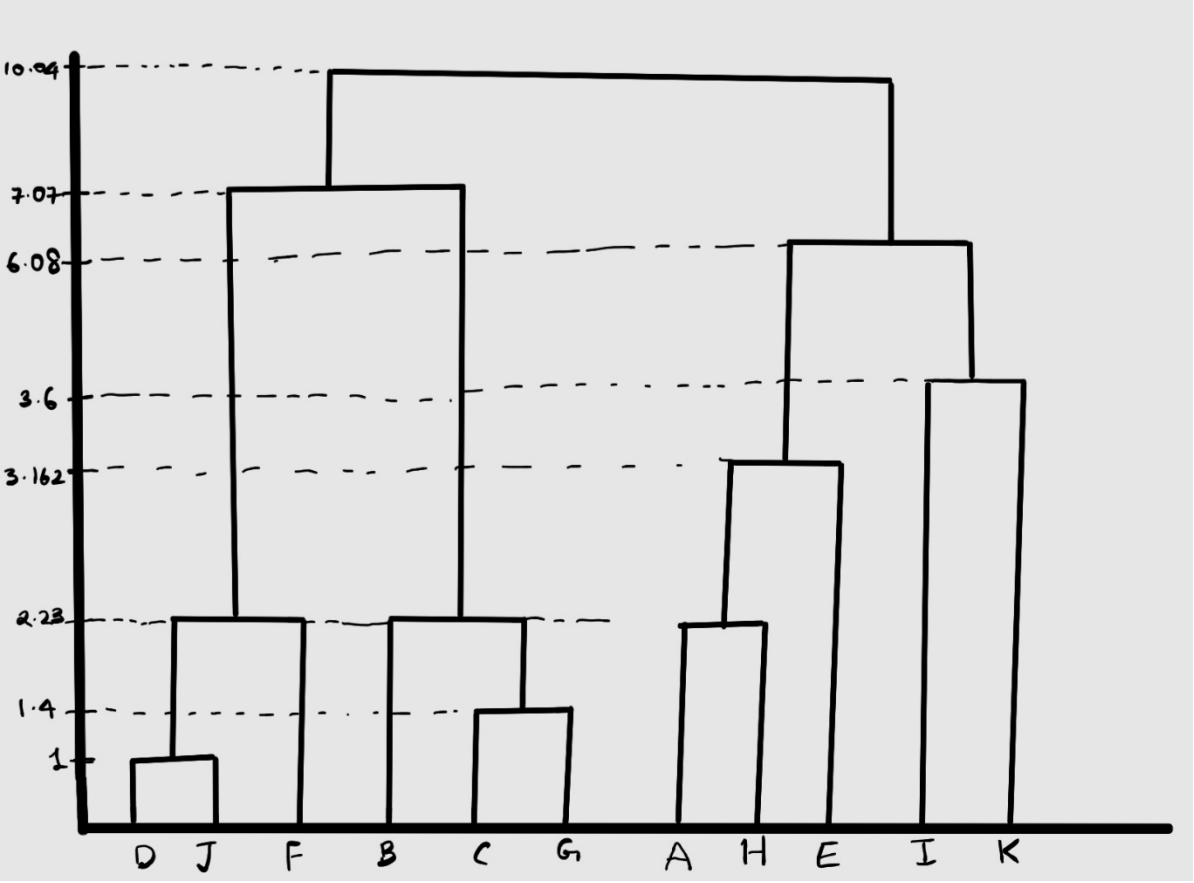
ROUND 9:

	A,E,H,I,K	B,C,G	D,F,J
A,E,H,I,K	0		
B,C,G	7.280109889	0	
D,F,J	10.04987562	7.071068	0

ROUND 10:

	A,B,C,E,G,H,I,K	D,F,G
A,E,H,I,K	0	
B,C,D,G,F,J	10.04987562	0

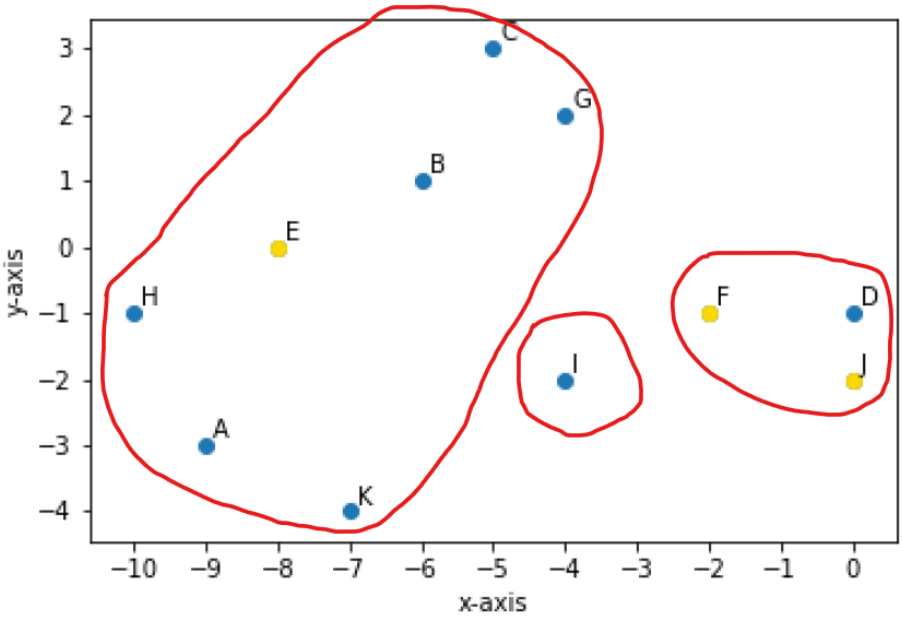
DENDOGRAM FOR COMPLETE LINK MATRIX



(b) (4 points) Using Sum of Squared Error (SSE) and assuming there are three clusters, which of the single link and complete link hierarchical clustering will yield better results? Justify your answer.

SINGLE LINK HIERARCHICAL CLUSTERING

We have obtained three clusters,
Cluster 1:ABCEGHK,
Cluster 2: I
Cluster 3: DFJ



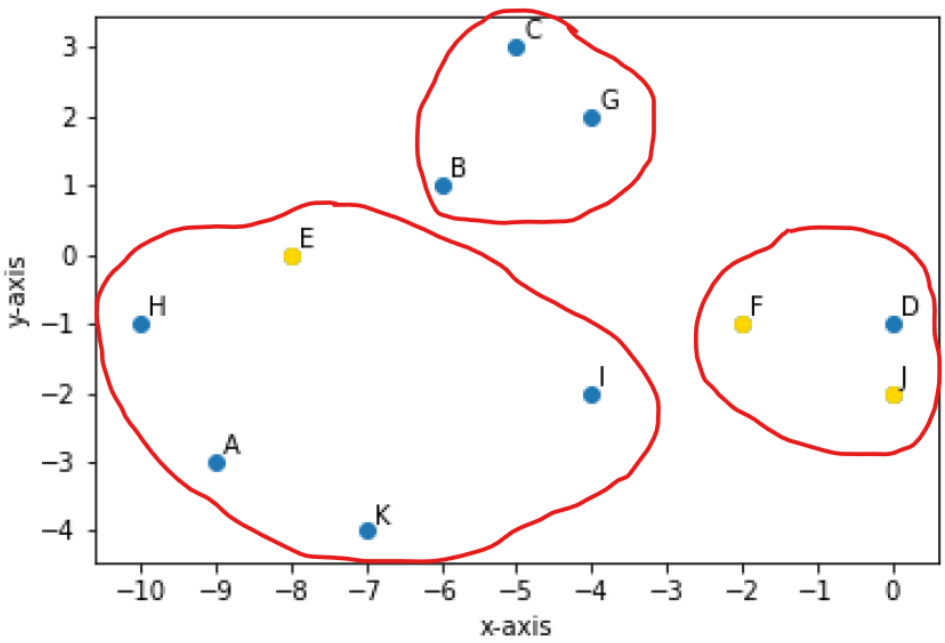

```python
import numpy as np
# Function to determine centroid of a cluster
def find_centroid(df):
  x_sum = df['x'].sum()
  y_sum = df['y'].sum()
  centroid = [x_sum/len(df), y_sum/len(df)]
  return centroid

# Function to find Sum of Squared Error
def SSE(cluster,cent):
  err = 0
  cent1 =  np.array(cent)
  x = euclidean_distances(cluster,cent1.reshape(1,-1)) ** 2
  for val in x:
    err += val
  return err
```

```
# SSE for Single Link cluster
cluster_1 = df.iloc[[0,1,2,4,6,7,10]] # ABCEGHK
cluster_2 = df.iloc[[8]] # I
cluster_3 = df.iloc[[3,5,9]] # DFJ

cent_1 = find_centroid(cluster_1)
cent_2 = find_centroid(cluster_2)
cent_3 = find_centroid(cluster_3)

print(cent_1,cent_2,cent_3)

print("Cluster 1 SSE: ", SSE(cluster_1,cent_1))
print("Cluster 2 SSE: ", SSE(cluster_2,cent_2))
print("Cluster 3 SSE: ", SSE(cluster_3,cent_3))

print("Total Error  : ", SSE(cluster_1,cent_1) + SSE(cluster_2,cent_2) + SSE(cluster_3,cent_3))
```

```
[-7.0, -0.2857142857142857] [-4.0, -2.0] [-0.6666666666666666, -1.3333333333333333]
Cluster 1 SSE:  [67.42857143]
Cluster 2 SSE:  [0.]
Cluster 3 SSE:  [3.33333333]
Total Error  :  [70.76190476]
```

COMPLETE LINK HIERARCHICAL CLUSTERING

We have obtained three clusters,
Cluster 1:AEHIK
Cluster 2: BCG
Cluster 3: DFJ



```
# SSE for complete Link cluster
cluster_1 = df.iloc[[0,4,7,8,10]] #AEHIK
cluster_2 = df.iloc[[1,2,6]] # BCG
cluster_3 = df.iloc[[3,5,9]] #DFJ

cent_1 = find_centroid(cluster_1)
cent_2 = find_centroid(cluster_2)
cent_3 = find_centroid(cluster_3)

print(cent_1,cent_2,cent_3)

print("Cluster 1 SSE: ", SSE(cluster_1,cent_1))
print("Cluster 2 SSE: ", SSE(cluster_2,cent_2))
print("Cluster 3 SSE: ", SSE(cluster_3,cent_3))

print("Total Error  : ", SSE(cluster_1,cent_1) + SSE(cluster_2,cent_2) + SSE(cluster_3,cent_3))
```

```
[-7.6, -2.0] [-5.0, 2.0] [-0.6666666666666666, -1.3333333333333333]
Cluster 1 SSE:  [31.2]
Cluster 2 SSE:  [4.]
Cluster 3 SSE:  [3.33333333]
Total Error  :  [38.53333333]
```

From the script above, we notice that complete link hierarchical clustering has a lesser SSE rate of **38.53** compared to single link's **70.76**.

(c) (3 points) Compare the clusters from 2(b) with the clusters found using K-means in Question 1 by calculating their corresponding Sum of Squared Errors (SSE)s. According to their SSE results, which is better: K-means or hierarchical clustering?

```
# SSE for K-Means
cluster_1 = df.iloc[[0,4,7,10]] # AEHK
cluster_2 = df.iloc[[1,2,6,8]] # BCGI
cluster_3 = df.iloc[[3,5,9]] # DFJ

cent_1 = find_centroid(cluster_1)
cent_2 = find_centroid(cluster_2)
cent_3 = find_centroid(cluster_3)

print(cent_1,cent_2,cent_3)

print("Cluster 1 SSE: ", SSE(cluster_1,cent_1))
print("Cluster 2 SSE: ", SSE(cluster_2,cent_2))
print("Cluster 3 SSE: ", SSE(cluster_3,cent_3))

print("Total Error  : ", SSE(cluster_1,cent_1) + SSE(cluster_2,cent_2) + SSE(cluster_3,cent_3))
```

```
[-8.5, -2.0] [-4.75, 1.0] [-0.6666666666666666, -1.3333333333333333]
Cluster 1 SSE:  [15.]
Cluster 2 SSE:  [16.75]
Cluster 3 SSE:  [3.33333333]
Total Error  :  [35.08333333]
```

Although hierarchical clustering is supposed to perform better on smaller datasets, we observe that K-Means clustering performs better in this case with an SSE of **35.0833 < 38.533 (Hierarchical Complete Clustering).** Thus, K-Means is the better suited for this dataset.

Q3: ***(8 points) [Frequent Itemset] [Angela Zhang (Designed) & Tyrone Wu (Graded)] For the transaction Table 4 given below, please answer the following questions:***

TID	Items Bought
T1	{C, D, E, G, H}
T2	{A, C, D, F}
T3	{A, C, E, F, G, H}
T4	{A, B, C, G}
T5	{D, E, F, H}
T6	{A, H}
T7	{A, B, C, F}
T8	{A, B, D, F, G}
T9	{A, B, E, G}
T10	{C, D, F, H}
T11	{A, B, F, H}
T12	{C, F, H}
T13	{A, C, D, E}
T14	{B, C, E, F, G}
T15	{A, C, F, H}

a) (1 point) Explain what is a frequent itemset and give an example of a 2-itemset that is a frequent itemset with minimal support count = 7.

Solution:

Frequent itemset are those itemsets where the support is ≥ the minimum support threshold.
{C,F} is an example of 2-itemset with a minimal support count of 7.

b) ***(3 points) Explain what is a closed frequent itemset and list ALL of them with support count = 7 in alphabetical order. No partial credit.***

Solution:

A frequent itemset is called a closed frequent itemset if none of its immediate supersets has the same support as the itemset.

The closed frequent itemset with minimum support as 7 are as follows:

{A},
{C},
{F},
{H},
{C,F}

c) ***(3 points) Explain what is the maximal frequent itemset and list ALL of maximal itemset with support count = 7 in alphabetical order. No partial credit.***

An itemset is considered to be "maximal frequent" when none of its immediate supersets is frequent.

Solution:

The maximal frequent itemsets for the transaction having a support ≥ 7 are:
{A},
{H},
{C,F}

d) *(1 point) Compute the support and confidence for the association rule {A, C} → {F}.*

Solution:

We have a total of 15 transactions.

{A,C,F} appears in 4 of these transactions.

Support({A,C,F}) =$4/15 \ = \ 0.2666$

Confidence({A,C,F}) = $4/6 \ = \ 0.667$

4. (13 points) [Association Analysis] [Ge Gao (Designed) & TyroneWu (Graded)]. Consider the following market basket transactions shown in the Table 2 below.

Transaction ID	Items ordered
1	{Beef, Butter, Chicken}
2	{Butter, Chicken, Milk}
3	{Butter, Cheese, Eggs, Soda}
4	{Beef, Eggs, Juice, Milk, Soda}
5	{Beef, Eggs, Soda}
6	{Beef, Bread, Cheese, Eggs}
7	{Bread, Soda}
8	{Beef, Cheese, Chicken}
9	{Chicken, Juice, Soda}
10	{Cheese, Milk, Soda}

Table 2: Market Basket Transactions Data

(a) (2 points) How many items are in this data set? What is the maximum size of itemsets that can be extracted from this data set (only including itemsets that have ≥ 1 support count)?

- Beef - 5
- Butter - 3
- Chicken - 4
- Milk - 3
- Cheese - 4
- Eggs - 4
- Soda - 6
- Juice - 2
- Bread - 2

The total number of items in the data set is - **9**.

For the given transactions, 1, 2, …. 9-item sets are possible, but most of them would have support count = 0.

Hence, the largest itemset possible from these transactions with support count ≥ 1 is -
{ Beef, Eggs, Juice, Milk, Soda }.

(b) (2 points) What is the maximum number of association rules that can be extracted from this data (including rules that have zero support)?

For a set of transactions having 'd' unique items, the total number of possible association rules can be found using the formula -

$$\mathbf{3^d - 2^{d+1} + 1}$$

It is seen above that, d = 9.

Hence, the maximum number of association rules that can be extracted from the data is :

$3^9 - 2^{9+1} + 1$

$= 3^9 - 2^{10} + 1$

$= 19{,}683 - 1{,}024 + 1$

$=$ **18,660**.

(c) (2 points) What is the maximum number of 3-itemsets that can be derived from this data set (including those have zero support)?

Since there are 9 unique items in the data provided, the maximum number of 3-itemsets possible is a simple combination expression - $\mathbf{{}^nC_r}$, where n is 9 and r is 3.

${}^9C_3 = (9!) / 3!*(9-3)!$

$= 84$.

The Maximum number of 3-itemsets that can be derived from this dataset is **84**.

(d) (3 points) Find an itemset (of size 2 or larger) that has the largest support.

These are the various itemsets and their support possible with the data:

- 2-Itemsets and their support counts

  { Beef, Butter } - 1
  { Beef, Chicken } - 2
  { Beef, Milk } - 1
  { Beef, Cheese } - 2
  { Beef, Eggs } - 3
  { Beef, Soda } - 2
  { Beef, Juice } - 1
  { Beef, Bread } - 1

  { Butter, Chicken } - 2
  { Butter, Milk } - 1
  { Butter, Cheese } - 1
  { Butter, Eggs } - 1
  { Butter, Soda } - 1

  { Chicken, Milk } - 1
  { Chicken, Cheese } - 1
  { Chicken, Soda } - 1
  { Chicken, Juice } - 1

  { Milk, Cheese } - 1
  { Milk, Eggs } - 1
  { Milk, Soda } - 2
  { Milk, Juice } - 1

  { Cheese, Eggs } - 2
  { Cheese, Soda } - 2
  { Cheese, Bread } - 1

  { Eggs, Soda } - 3
  { Eggs, Juice } - 1
  { Eggs, Break } - 1

  { Soda, Juice } - 2
  { Soda, Bread } - 1

- 3-itemsets and their support counts

  { Beef, Butter, Chicken } - 1
  { Butter, Chicken, Milk } - 1
  { Beef, Eggs, Soda } - 2
  { Beef, Cheese, Chicken } - 1
  { Chicken, Juice, Soda } - 1
  { Chess, Milk, Soda } - 1

- 4-itemsets and their support counts

  { Butter, Cheese, Eggs, Soda } - 1
  { Beef, Bread, Cheese, Eggs} - 1

- 5-itemsets and their support counts

  { Beef, Eggs, Juice, Milk, Soda } - 1

From all the itemsets above, **{ Beef, Eggs }** and **{ Eggs, Soda }** with **3**, have the highest confidence.

(e) (4 points) Find two pairs of items, a and b, such that the rules {a} → {b} and {b} → {a} have the same confidence, and the support for each rule is greater than or equal to 0.2.

Confidence of {a} → {b} is **σ(a,b) / σ(a)**

Confidence of {b} → {a} is **σ(a,b) / σ(b),** where σ is support-count.

For confidence of {a} → {b} to be equal to confidence of {b} → {a}:

σ(a,b) / σ(a) = σ(a,b) / σ(b)

Or

σ(a) = σ(b) …….. (Condition 1)

And

Support (S) must be greater than or equal to 02.

But since Support of {a} → {b} is - σ(a,b) / Total number of transactions, and there are 10 transactions.

Hence, the **support count** of the rule needs to be **≥ 2.** …… (Condition 2)

- Itemsets with support count ≥ 2:

  { Beef, Chicken } - 2
  { Beef, Cheese } - 2
  { Beef, Soda } - 2
  { Butter, Chicken } - 2
  { Milk, Soda } - 2
  { Cheese, Eggs } - 2
  { Cheese, Soda } - 2
  { Eggs, Soda } - 3
  { Soda, Juice } - 2
  { Beef, Eggs, Soda } - 2

Out of all these items sets,

{ Cheese, Eggs } is the one where { Cheese } has support count = 4 and { Eggs } has support count = 4.

Assuming a = Cheese and b = Eggs,

- It is true that σ(a) = σ(Cheese) is equal to σ(b) = σ(Eggs), with count = 4.
- And also for {Cheese} → {Eggs}, Support is 2/10 = 0.2.

Based on the 2 conditions defined above, **{a} and {b} can be {Cheese} and {Eggs} or {Eggs} and {Cheese}.**

So, {a} → {b} can be either **{Cheese}** → **{Eggs}** or **{Eggs}** → **{Cheese}**.

---

5. (24 points) [Apriori algorithm] [Ge Gao (Designed) & Pragna Bollam (Graded)]. Consider the data set shown in Table 3 and answer the following questions using Apriori algorithm.

TID	Items
t_1	A,B
t_2	A,B,E
t_3	A,B,D
t_4	B,E
t_5	A,B,D,E
t_6	C,D
t_7	B,C,D,F
t_8	A,D
t_9	A,B,D,F
t_{10}	A,B,C

Table 3: Apriori algorithm

(a) (10 points) Show (compute) each step of frequent itemset generation process using Apriori algorithm, with support count of 2.

Apriori Principle: This principle holds the condition that:

$$\forall\ X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

This translates to the condition that the Support of an item set never exceeds the support of its subset. Rather than checking support for larger item sets, the support of its subsets can be checked. If this subset support is less than 2, all its supersets can be pruned.

We use Apriori Algorithm to calculate itemsets and their respective support count:

Step-1: For 1-itemsets

Itemset	Support Count
{A}	7
{B}	8

{C}	3
{D}	6
{E}	3
{F}	2

Step-2: These are the 2-itemsets that have been generated from the 1-itemsets.The pruned itemsets are highlighted below.

Itemset	Support Count
{A,B}	6
{A,C}	1
{A,D}	4
{A,E}	2
{A,F}	1
{B,C}	2
{B,D}	4
{B,E}	3
{B,F}	2
{C,D}	2
{C,E}	0
{C,F}	1
{D,E}	1
{D,F}	2
{E,F}	0

From the 1-itemsets, these 2-itemsets have been generated. Similarly, 3-itemsets must be generated from these. But before that, the ones having a support count < 2 have been eliminated.

Step-3: 3-itemsets from the 2-itemsets that have not been pruned.

Itemset	Support Count
{A,B,D}	3
{A,B,E}	2
{A,D,E}	1
{B,C,D}	1
{B,C,E}	0
{B,C,F}	1
{B,D,E}	1
{B,D,F}	2
{B,E,F}	0

Step-4: 4-itemsets from the 3-itemsets that have not been pruned.

Itemset	Support Count
{A,B,D,E}	1

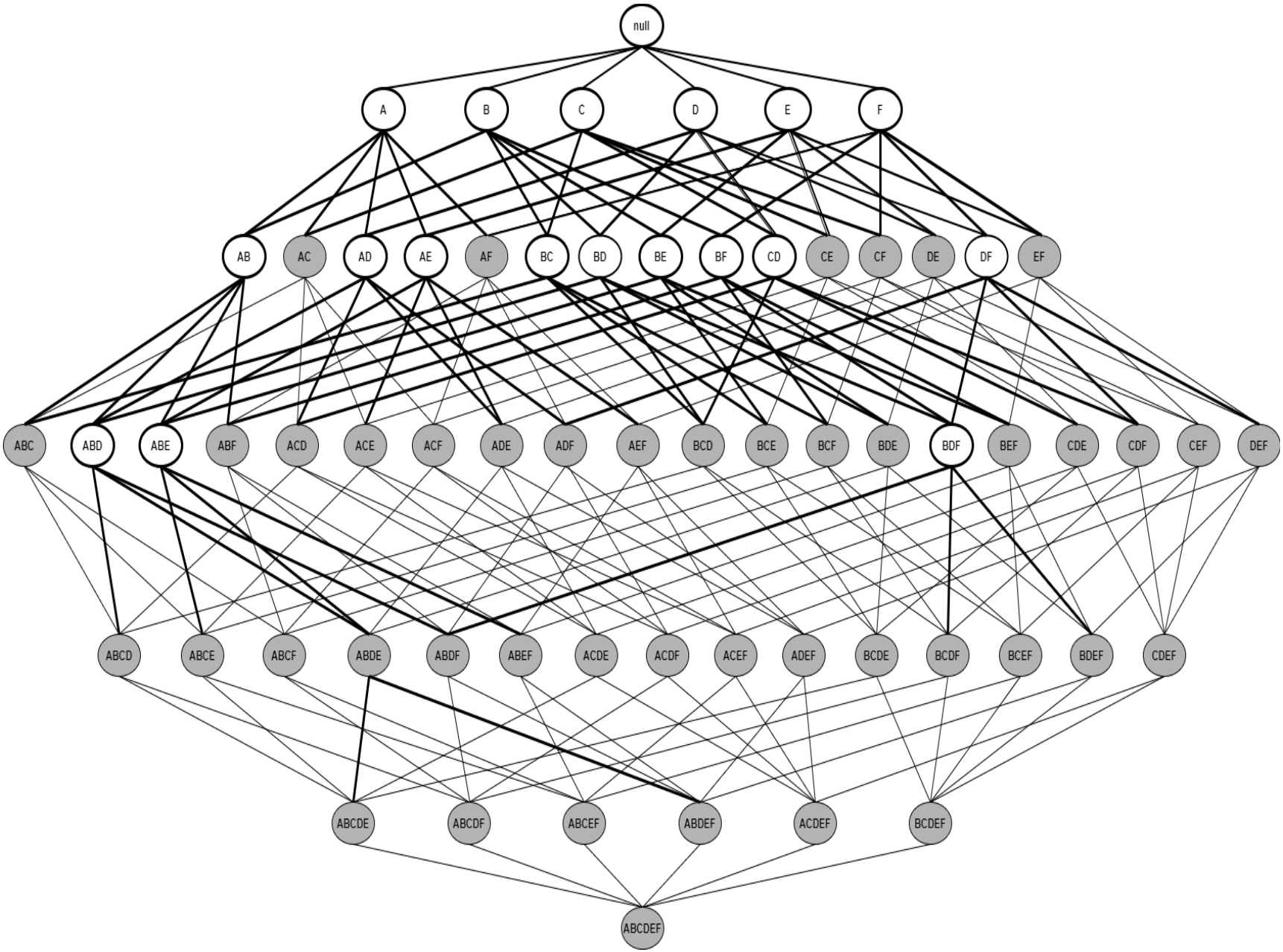
But {A,B,D,E} has support count less than 2.

Since more itemsets cannot be generated, **{A,B,D}**, **{A,B,E}** and **{B,D,F}** would be the frequent itemsets.

(b) (10 points) Show the lattice structure for the data given in table above, and mark the pruned branches if any. (Scanned hand-drawing is acceptable as long as it is clear.)

In the lattice structure below, the grey nodes are itemsets that were pruned directly, or their subsets were pruned for having a support count less than 2.

The thicker branches or the highlighted branches should be kept, while the others have to be pruned.


null
A
B
C
D
E
F
AB
AC
AD
AE
AF
BC
BD
BE
BF
CD
CE
CF
DE
DF
EF
ABC
ABD
ABE
ABF
ACD
ACE
ACF
ADE
ADF
AEF
BCD
BCE
BCF
BDE
BDF
BEF
CDE
CDF
CEF
DEF
ABCD
ABCE
ABCF
ABDE
ABDF
ABEF
ACDE
ACDF
ACEF
ADEF
BCDE
BCDF
BCEF
BDEF
CDEF
ABCDE
ABCDF
ABCEF
ABDEF
ACDEF
BCDEF
ABCDEF

(c) (4 points) Mark closed and maximal frequent itemsets on the lattice structure from (b) if there's any.

Closed frequent itemset: An itemset S is said to be a closed frequent itemset if not of its supersets have the same support as S.

For the data above, the Closed frequent itemset is - {A}, {B}, {C}, {D}, {A,B}, {A,D}, {B,C}, {B,D}, {B,E}, {C,D}, {A,B,D}, {A,B,E} and {B,D,F}.

Maximal frequent itemset: An itemset S is said to be a maximal frequent itemset if none of its immediate supersets are frequent.

For the data above, the Maximal frequent itemset is - {B,C}, {C,D}, {A,B,D}, {A,B,E} and {B,D,F}.

---

6. (30 points) [Frequent Pattern Tree] [Angela Zhang (Designed) & Chengyuan Liu (Graded)] Consider the following data set shown in Table 4 and answer the following questions using FP-Tree.

TID	Items Bought
T1	{C, D, E, G, H}
T2	{A, C, D, F}
T3	{A, C, E, F, G, H}
T4	{A, B, C, G}
T5	{D, E, F, H}
T6	{A, H}
T7	{A, B, C, F}
T8	{A, B, D, F, G}
T9	{A, B, E, G}
T10	{C, D, F, H}
T11	{A, B, F, H}
T12	{C, F, H}
T13	{A, C, D, E}
T14	{B, C, E, F, G}
T15	{A, C, F, H}

Table 4: Transactions Data

(a) (15 points) Construct an FP-Tree for the set of transactions in the table below as the first step towards identifying the itemsets with a minimum support count of 2 (at least 2 occurrences). Hint: Do not forget to include the header table that locates the starts of the corresponding linked item lists through the FP-Tree.

The items and their count from the data above is:

Item	Support Count
A	10
B	6
C	10
D	6

E	6
F	10
G	6
H	8

Sorting the items based on their frequency to generate FP-Tree.

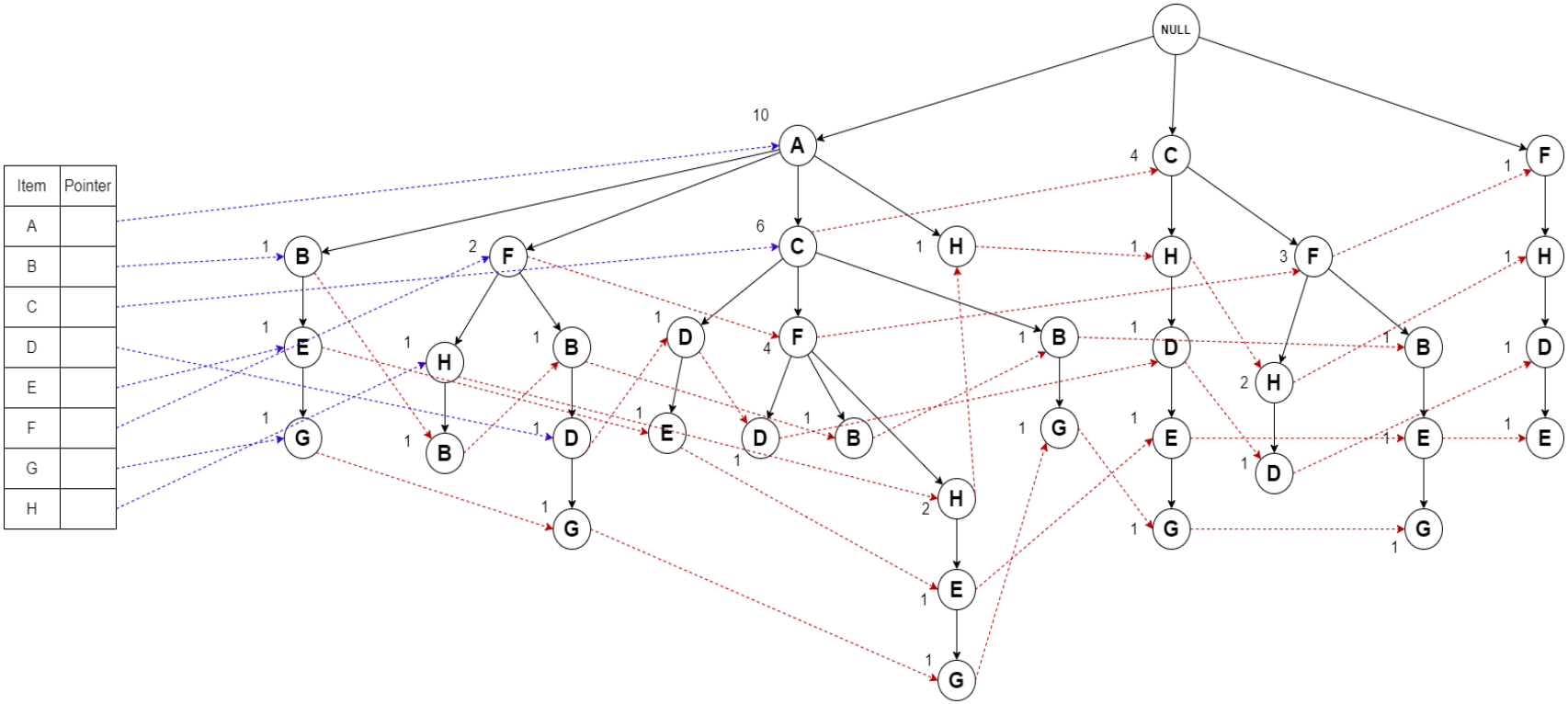
Item	Support Count
A	10
C	10
F	10
H	8
B	6
D	6
E	6
G	6

Now the transactional data must be ordered such that the most frequent items occur first.

TID	Items Bought	Items Bought re-ordered
T1	{C,D,E,G,H}	{C,H,D,E,G}
T2	{A,C,D,F}	{A,C,F,D}
T3	{A,C,E,F,G,H}	{A,C,F,H,E,G}
T4	{A,B,C,G}	{A,C,B,G}
T5	{D,E,F,H}	{F,H,D,E}
T6	{A,H}	{A,H}

T7	{A,B,C,F}	{A,C,F,B}
T8	{A,B,D,F,G}	{A,F,B,D,G}
T9	{A,B,E,G}	{A,B,E,G}
T10	{C,D,F,H}	{C,F,H,D}
T11	{A,B,F,H}	{A,F,H,B}
T12	{C,F,H}	{C,F,H}
T13	{A,C,D,E}	{A,C,D,E}
T14	{B,C,E,F,G}	{C,F,B,E,G}
T15	{A,C,F,H}	{A,C,F,H}

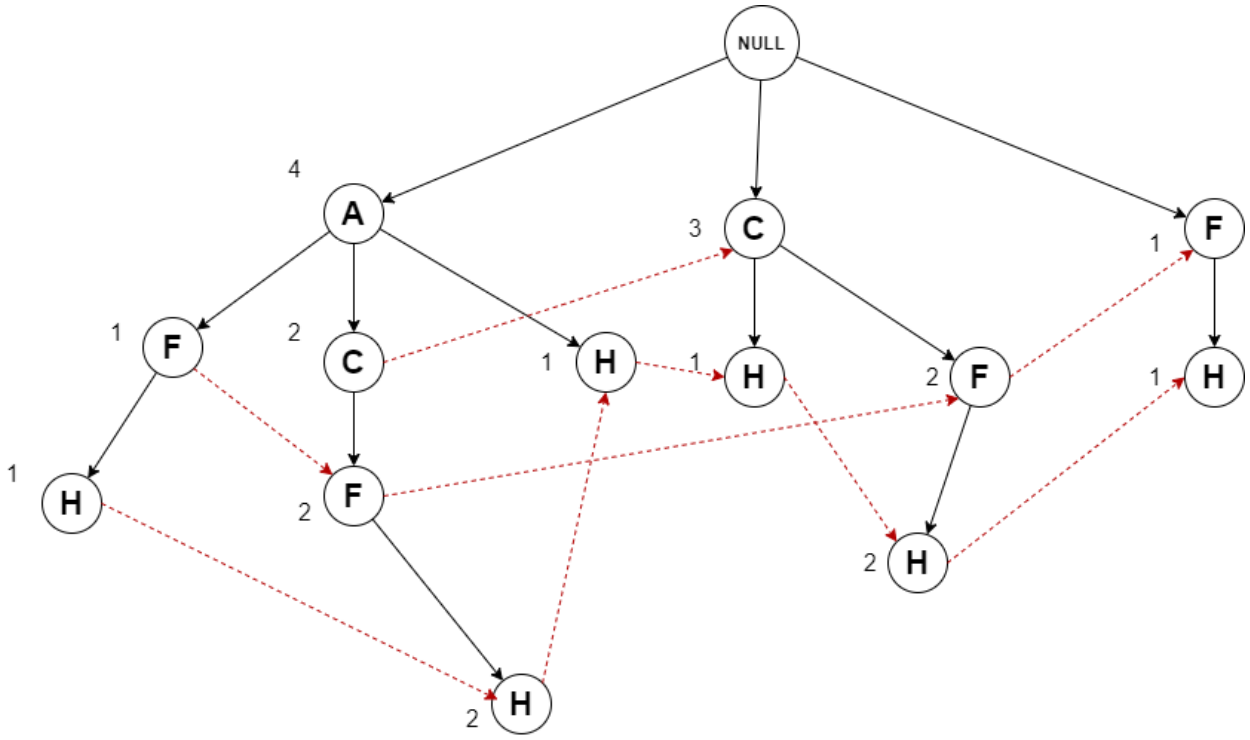
Based on the re-ordered item sets the FP-Tree is built.


Item
Pointer
A
B
C
D
E
F
G
H
NULL

(b) (15 points) Using the FP-Tree constructed and support count = 2, generate all the frequent patterns with the base of item H step by step. List the frequent itemsets in alphabetical order.

To understand and solve for the frequent patterns on the data with the base item as H, a subtree having all paths from the root to the node H needs to be generated. Such subtree is called the prefix path subtree for H.

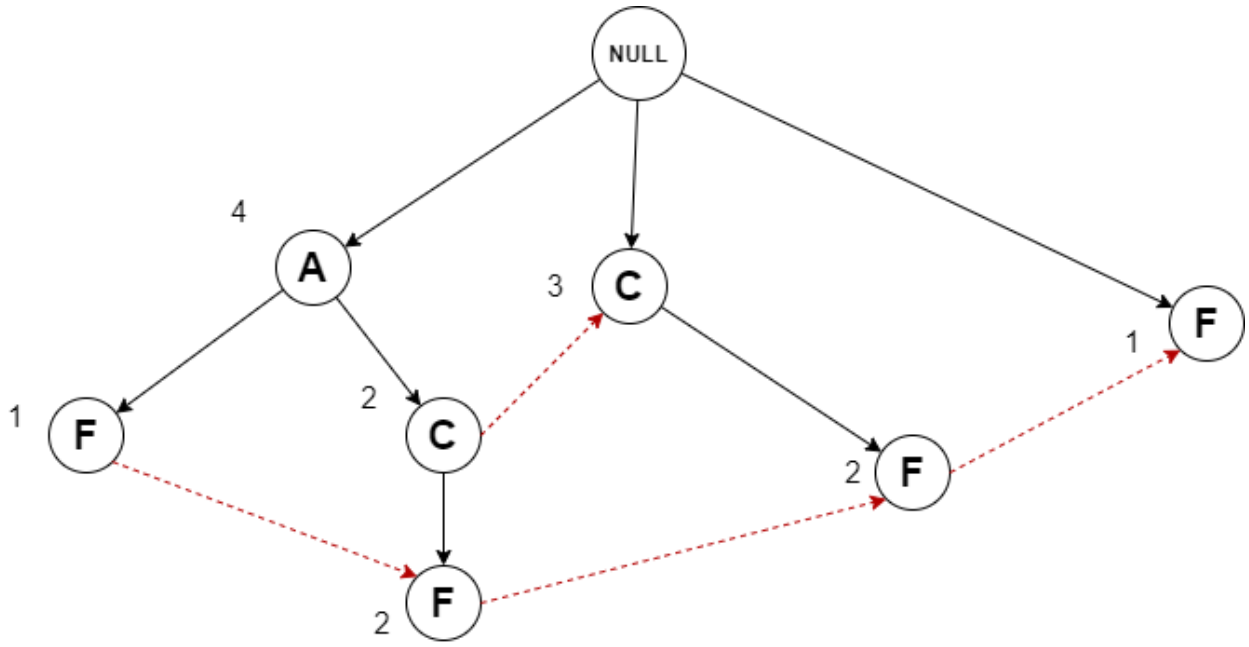
The subtree of H is:



To find the frequent patterns for the given data these steps must be followed:

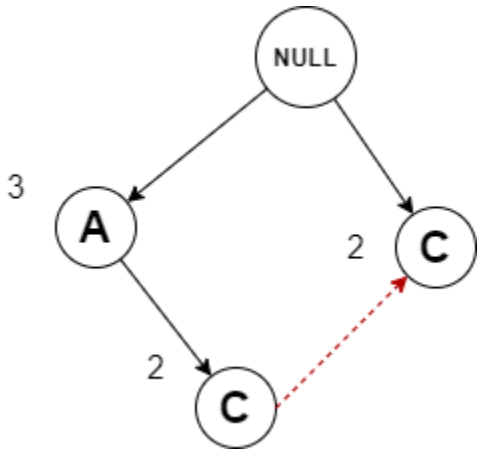
- Build the prefix path from the base item, i.e, H.
- Check if H is a frequent item, if so extract it.
- If H is frequent, find its subsets and do:
  :
    - Break the patterns into smaller subsets.
    - Solve for each subset.

This is the conditional FP-tree for **H**. Based on the condition that the support count must be equal or greater than 2, frequent patterns can be found.



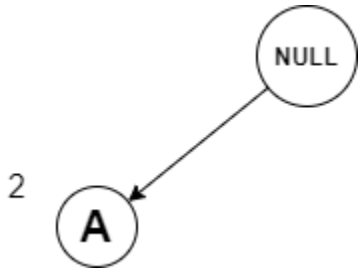
As observed from above, the other items - A, F, and C are frequent with support counts ≥ 2. So similarly, conditional FP-trees of AH, FH, and CH should also be generated.

1. The conditional FP-tree for **FH** is:



But both C and AC are frequent with support count ≥ 2, the same process is repeated for CFH and ACFH.

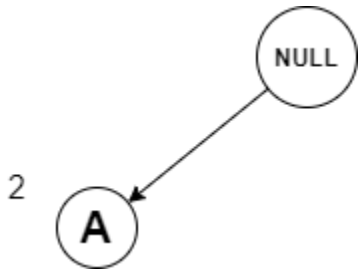
The conditional FP-tree for **CFH** is:



The conditional FP-tree for ACFH is:

Since both ACFH and CFH are frequent, the subset **FH must also be frequent**.

2. The conditional FP-tree for **CH** is:



Since A is frequent with support count ≥ 2, the same process repeats for ACH.

Conditional FP-tree for ACH is:

CH is also frequent.

3. The conditional FP-tree for **AH** is:

From all the subtrees, it can be said that these are the frequent patterns of the data :

Itemset	Support count
H	8
AH	4
CH	5
FH	6
CFH	4
ACFH	2
AFH	3
ACH	2
