DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

HW2

Abhinav Barnawal *Shreyansh Singh* *Si Siddhanth Raja*
: *2020CS50415* *2020CS10385* *2020CS50443*
Class: *2301-COL761*
Session: *2023-24*
: *cs5200415@iitd.ac.in* *cs1200385@iitd.ac.in* *cs5200443@iitd.ac.in*

Course: *Data Mining* – Instructor: *Prof. Sayan Ranu*
Submission date: *October 23, 2023*

Q1

We ran all three algorithms downloaded from the links in the assignment document. Due to the unavailability of HPC and limited RAM on BADAAL, I have run the script on Google Colab; therefore, the results may vary slightly. However, to reproduce the result, please go into the directory Q1 containing script.sh. To run a dataset, say "167.txt_graph" in the same format as the yeast dataset, run the script using the command:

```
bash script.sh 167.txt_graph
```

We obtained the following plot:

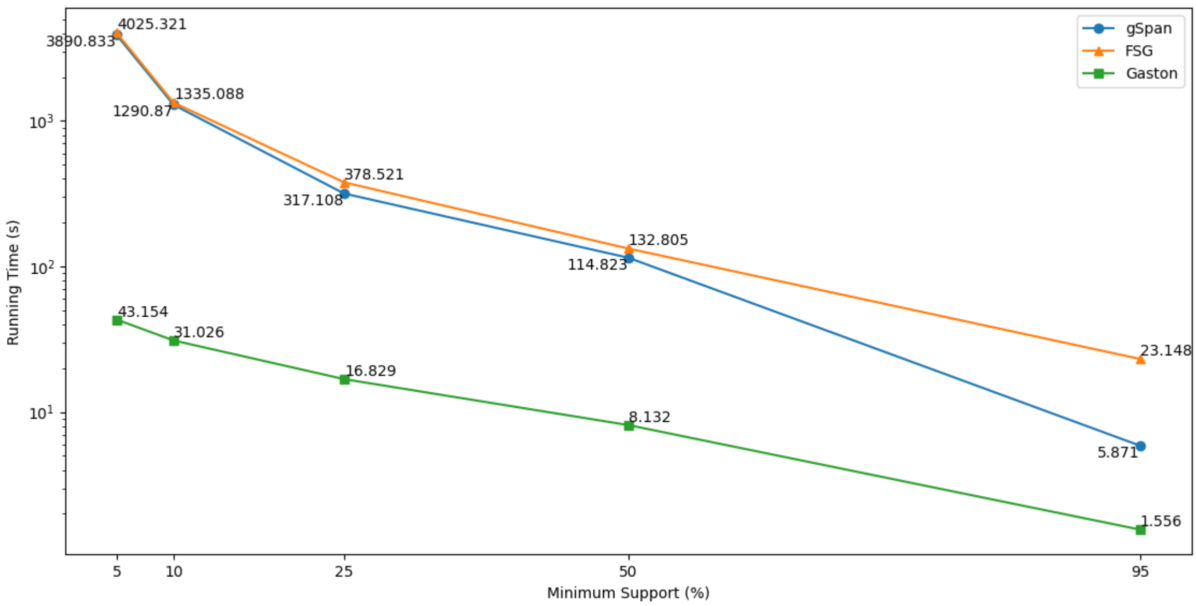


Figure 1: Showing Total Running Times of the three algorithms on yeast dataset

We observe that the running time decreases almost exponentially (linearly on log scale) with increasing minSup value for all the algorithms because the low-frequency subgraphs are not generated with higher minSup. Though the growth rate of gaston is much less than the other two, while the growth rate of gSpan is marginally larger than that of FSG. Further, gaston is the fastest, followed by gSpan, and FSG is the slowest algorithm irrespective of the value of minimum-support. This happens because:

1. gSpan strictly avoids infrequent candidate generations, which is an expensive step in the FSG algorithm. gSpan also builds a new lexicographic order among graphs, further pruning the search space and reducing expensive graph-isomorphism tests.

2. gaston is faster than the gSpan algorithm because gaston uses an entirely different approach to generate and prune the candidate subgraphs. It uses the quick-start principle, i.e., it first considers only simple paths as potential candidates. It then merges them to consider complex trees, further extended to complex cyclic graphs. gaston benefits from the fact that most frequent subgraphs are not too complex and can be discovered cost-effectively. This also makes gaston manifold faster than the other two approaches, while the difference between gSpan and FSG is not as much.

Q2

Elbow plot for K-means Clustering

We have used the KMeans library from sklearn. Then, we calculated the sum of the squared distance for each point from its cluster center for k ranging from 1 to 15 and stored them in a list. For finding the elbow in the plot, we check if the difference between any two consecutive values is less than the threshold (threshold = 1). The value of k corresponding to this value is our optimal k. The value of the sum of squared distances decreases as the k increases, and after some time, the graph becomes almost parallel to the x-axis, and the optimal k is the k, after which the graph becomes flat. So, for determining the optimal k, we have selected 1 as our threshold, which is decided after observing the graphs for different dimensions. This threshold value will work for all the dimensions from [4, 7].

To generate the plot:

```
sh elbow_plot.sh CS1200385_generated_dataset_7D.dat 7 q2_7_CS1200385.png
```

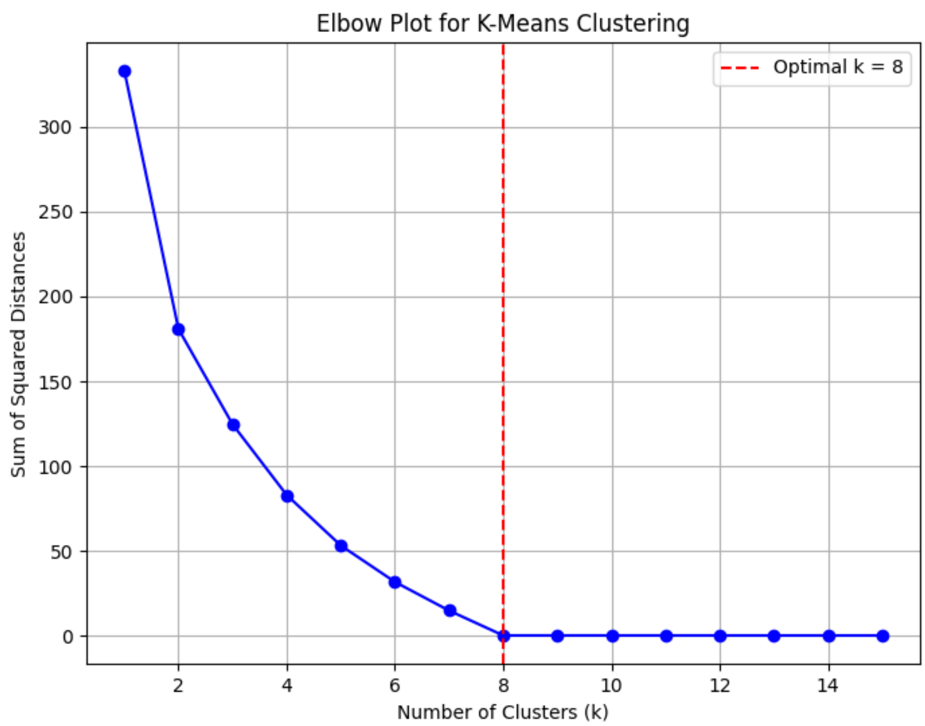


Figure 2: Elbow plot for K-Means Clustering

Q3

(1)

Draw the dendrogram for single linkage clustering on the data below. Show all the steps.[**5 marks**]

Solution :

1. Compute Distance Matrix -

$$\begin{bmatrix} Clusters & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ v_1 & 0 & 0.234 & 0.21 & 0.36 & 0.34 & 0.235 \\ v_2 & 0.234 & 0 & 0.143 & 0.194 & 0.143 & 0.243 \\ v_3 & 0.21 & 0.143 & 0 & 0.158 & 0.2846 & 0.101 \\ v_4 & 0.36 & 0.194 & 0.158 & 0 & 0.2842 & 0.219 \\ v_5 & 0.34 & 0.143 & 0.2846 & 0.2842 & 0 & 0.386 \\ v_6 & 0.235 & 0.243 & 0.101 & 0.219 & 0.386 & 0 \end{bmatrix}$$

2. Find the smallest distance pair and merge the clusters.

   (a) Merge v_3 and v_6

$$\begin{bmatrix} Clusters & v_1 & v_2 & v_3, v_6 & v_4 & v_5 \\ v_1 & 0 & 0.234 & 0.21 & 0.36 & 0.34 \\ v_2 & 0.234 & 0 & 0.143 & 0.194 & 0.143 \\ v_3, v_6 & 0.21 & 0.143 & 0 & 0.158 & 0.2846 \\ v_4 & 0.36 & 0.194 & 0.158 & 0 & 0.2842 \\ v_5 & 0.34 & 0.143 & 0.2846 & 0.2842 & 0 \end{bmatrix}$$

   (b) Merge v_2 and v_5

$$\begin{bmatrix} Clusters & v_1 & v_2, v_5 & v_3, v_6 & v_4 \\ v_1 & 0 & 0.234 & 0.21 & 0.36 \\ v_2, v_5 & 0.234 & 0 & 0.143 & 0.194 \\ v_3, v_6 & 0.21 & 0.143 & 0 & 0.158 \\ v_4 & 0.36 & 0.194 & 0.158 & 0 \end{bmatrix}$$

   (c) Merge $\{v_2,\ v_5\}$ and $\{v_3,\ v_6\}$

$$\begin{bmatrix} Clusters & v_1 & v_2, v_5, v_3, v_6 & v_4 \\ v_1 & 0 & 0.21 & 0.36 \\ v_2, v_5, v_3, v_6 & 0.21 & 0 & 0.158 \\ v_4 & 0.36 & 0.158 & 0 \end{bmatrix}$$

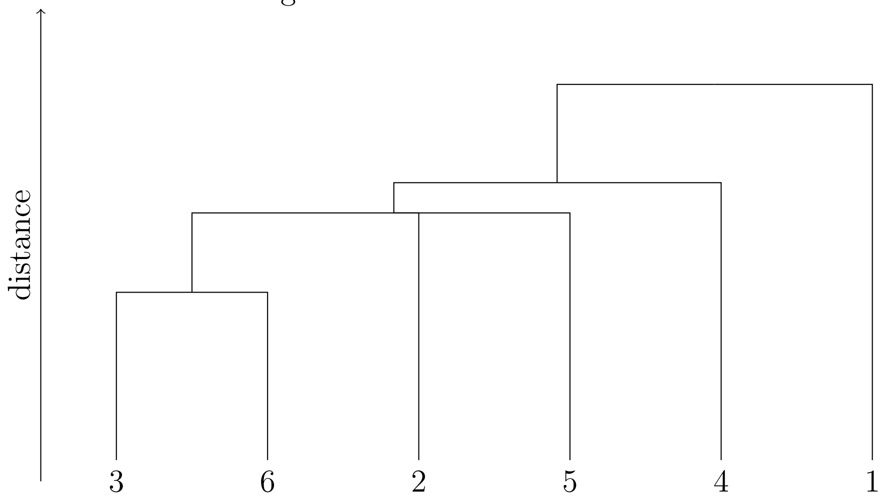
   (d) Merge $\{v_2,\ v_5,\ v_3,\ v_6\}$ and v_4

$$\begin{bmatrix} Clusters & v_1 & v_2, v_5, v_3, v_6, v_4 \\ v_1 & 0 & 0.21 \\ v_2, v_5, v_3, v_6, v_4 & 0.21 & 0 \end{bmatrix}$$

   (e) Merge v_1 and $\{v_2,\ v_5,\ v_3,\ v_6,\ v_4\}$

$$\begin{bmatrix} Clusters & v_1, v_2, v_5, v_3, v_6, v_4 \\ v_1, v_2, v_5, v_3, v_6, v_4 & 0 \end{bmatrix}$$

The resultant Dendogram -



(2)

What is the complexity of the fastest possible algorithm? Give your algorithm's pseudocode and complexity analysis. **[10 marks]**

Solution :
Single Linkage Clustering can be done in $\mathbf{O(n^2)}$ time using the SLINK Algorithm.

Algorithm 1: Pre-Processing

```
Data: Data (v_1, v_2, ..., v_n), Distance Function dist
Result: Distance Matrix D, A_v, A_d
/* D is the distance matrix                                             */
D ← {0}_{n×n};
/* A_v stores the neighbor with the min inter-cluster distance for each
   cluster (single point initially) and A_d stores the corresponding
   distance                                                             */
A_v ← {−1}_n;
A_d ← {∞}_n;
for i ← 1 to n do
    minD ← ∞;
    minV ← −1;
    for j ← 1 to n do
        D[i][j] = dist(v_i, v_j);
        if D[i][j] ≤ minD & j ≠ i then
            minD = D[i][j];
            minV = j;
        end
    end
    A_v[i] = minV;
    A_d[i] = minD;
end
return (D, A_v, A_d);
```

Algorithm 2: Single Link Clustering

```
Data: Data (v_1, v_2, ..., v_n), D, A_v, A_d
Result: Dendogram DG
/* DG stores the clusters (set) at each level of the Dendogram */
DG ← {}_n;
DG[n] = {{v_1}, {v_2}, ... {v_n}};
for level ← n − 1 to 1 do
    /* Find the clusters i and j with the min inter-cluster distance */
    d ← ∞;
    i ← −1;
    for k ← 1 to n do
        if A_d[k] < d then
            d = A_d[k];
            i = k;
        end
    end
    j = A_v[i];
    if i < j then
        /* Merge i and j */
        minD ← ∞;
        minV ← −1;
        for k ← 1 to n do
            /* Update the Distance Matrix after merger */
            D[i][k] = D[k][i] = min(D[i][k], D[j][k]);
            D[j][k] = D[k][j] = ∞;
            if k ≠ i & k ≠ j then
                /* Update A_v and A_d with new min intra-cluster distances */
                if A_v[k] == i || A˙v[k] == j then
                    A_v[k] = i;
                    A_d[k] = min(D[k][i], D[k][j]);
                end
                if D[i][k] ≤ minD then
                    minD = D[i][k];
                    minV = k;
                end
                if D[j][k] ≤ minD then
                    minD = D[j][k];
                    minV = k;
                end
            end
        end
        A_v[i] = minV;
        A_d[i] = minD;
        A_v[j] = −1;
        A_d[j] = ∞;
        /* Append the clusters to DG */
        DG[level] = DG[level + 1];
        DG[level][i] = DG[level][i] ∪ DG[level][j];
        DG[level][j] = {};
    end
end
return DG;
```

Time Complexity Analysis

Pre-processing 1 takes $\mathbf{O(n^2)}$ time to compute the Distance Matrix.
Clustering 2 involves $\mathbf{O(n)}$ computations for each of the n levels of the Dendogram, taking overall $\mathbf{O(n^2)}$ time.

$$\therefore\ \mathcal{T}(n) = \mathbf{O(n^2)}$$

Acknowledgement

Our sincere thanks to Prof. Sayan Ranu for his expertise and guidance that led to the completion of this work. We also thank the respected Teaching Assistants for their immense support and almost instant replies to our doubts, without whom it would have never been possible to present this work.