CSE454 - HW2 180104617
Elifnur Kabalci

a) The Chameleon clustering algorithm is an approach used in data mining to cluster large databases. It's known for its adaptability, much like a chameleon, in identifying clusters with varying shapes, sizes and densities.
Two-phase approach:

Dynamic Modelling: Chameleon uses a graph-partitioning algorithm to partition data into a large # of small subclusters. This is done based on k-NN connectivity.

Cluster Merging: These subclusters are merged iteratively based on two criteria - relative closeness and relative interconnectivity.

Relative Closeness (RC): This measures how close two clusters are. It's calculated based on the intra-cluster edge weight.

Relative Interconnectivity (RI): This measures how interconnected two clusters are, based on the inter-cluster edge weight.

Pseudo Codes:

⇒ Create Subclusters:

```
for each point in the dataset:
    find its k-nearest neighbors
    Create edges in the graph
Use graph partitioning to divide the graph into subclusters
```

⇒ Merge Subclusters.

```
while there are subclusters to merge:
    Calculate RC and RI for each pair of subclusters
    If RC and RI are above predefined thresholds:
        Merge the subclusters
```

1) Relative Closeness (RC) formula:

$$RC = \frac{\text{Intra-cluster similarity}}{\text{Average intra-cluster similarity of two clusters}}$$

2) Relative Interconnectivity (RI) formula:

$$RI = \frac{\text{Inter-cluster similarity}}{\text{Average inter-cluster similarity of two clusters}}$$

b)

Advantages:

1) Dynamic Modelling: It adapts to the inherent data distribution, capable of identifying clusters with varying shapes and sizes.

2) Handling non-Uniform Densities: Excellently manages datasets with clusters of different densities.

3) Scalability: Good scalability for large datasets, particularly when implemented efficiently.

4) Hierarchical Clustering Approach: Offers a detailed view of data structure through its hierarchical nature.

Disadvantages:

1) Complexity and Overhead: Its more complex and involves higher computational overhead compared to simpler methods like k-means.

2) Sensitivity to Parameters: The performance is sensitive to its parameters like the k-NN and merging thresholds.

3) Not Ideals for High Dimensions: Performance can degrade with very high-dimensional data.

Comparisons:

	Time Complexity	Performance	Memory Consumption
k-Means	generally faster due to its simplicity, especially for $O(n)$.	Struggles with non spherical clusters, unlike Chameleon.	Generally lower than Chameleon, especially in standart implement.
DBSCAN	has higher time complexity, often $O(n^2)$, but handles varying density clusters better than k-means.	like Chameleon, is good with arbitrary shaped clusters but may struggle with varying density clusters	like Chameleon, especially in handling noise and outlier detection.
Hierarchical Clustering	often slower than Chameleon due to its complexity, especially in agglomerative approaches.	Provides a hierarchical view of clusters, similar to Chameleon, but can be less effient in handling varying densities.	High, particularly because it requires the storage of a distance matrix.
Spectral Clustering	can be computationally intensive, particularly in eigenvalue decomposition.	Excellent in identifying complex structures, similar to Chameleon but can be sensitive to the choice of similarity metrics.	High, especially in the computation of the similarity matrix.

c)

1) Building the k-NN graph:

Process: Each point in the dataset is connected to its k-nn.

Time Complexity: If using a brute-force approach, finding the knn for each point has a complexity of $O(n^2)$.

2) Partitioning the dataset:

Process: The dataset is partitioned into a large # of relatively small sub-clusters. This is typically done using a graph-partitioning algorithm.

Time Complexity: Most graph partitioning algorithms, like the Multilevel Recursive-Bisection used in Chameleon, have a complexity of $O(n \log n)$ or $O(n)$.

3) Merging Process:

Process: Pair of sub-clusters are merged based on relative interconnectivity and relative closeness. This step is iterative.

Time Complexity: The complexity can be approximated as $O(n^2)$, where n is the # of sub-clusters. Since in the worst case, each pair of sub-clusters might need to be evaluated.