CS4378V: Homework #1

(100 points)

CS4378V: Homework #1

1. (40 pt) The k-means algorithm is pretty straight forward. Here is the pseudo-code for it. Please implement k-means on 2-dimensional numerical data by C++, which should be fairly easy to derive from this.

```
  Let k be the number of clusters you want
  Let S be the set of data samples (|S| is the size of the set)
  Let A be the set of associate clusters for each data sample
  Let sim(x,y) be the similarity function
  Let c[k] be the vectors for cluster centers

  Init:
       Let S' = S
       //choose k random vectors to start our clusters
      for i=1 to k
            j = rand(|S'|)
           c[k] = S'[j]
          S' = S' - {c[n]} //remove that vector from S' so we can't choose it again
      end

    //assign initial clusters
      for i=1 to |S|
         A[i] = argmin(j = 1 to k) { sim(S[i], c[j]) }
      end

Run:
     Let change = true
    while change
          change = false //assume there is no change
         //reassign feature vectors to clusters
        for i = 1 to |S|
             a = argmin(j = 1 to k) { sim(S[i], c[j]) }
        if a != A[i]
            A[i] = a
            change = true //a vector changed affiliations -- so we need to
            //recompute our cluster vectors and run again
        end
   end

//recalculate cluster locations if a change occurred
 if change
      for i = 1 to k
      mean, count = 0
      for j = 1 to |S|
           if A[j] == i
                mean = mean + S[j]
               count = count + 1
     end
```

```
        end
        c[i] = mean/count
      end
    end
```

Use your code to cluster the following eight points (with (x, y) representing locations) into three clusters A1(2, 10) A2(2, 5) A3(8, 4) A4(5, 8) A5(7, 5) A6(6, 4) A7(1, 2) A8(4, 9). The distance (similarity) function between two points *a=(x1, y1)* and *b=(x2, y2)* is defined as: $\rho(a, b) = |x2 - x1| + |y2 - y1|$.

To be simple, you can just set k=3 and set S equals to the above eight points. Your program needs output the final clustering result (members in each cluster).
For example: Cluster1: {A1, A4, …}
Cluster2: {A3, A5, ..}
Cluster3: {A2, …}
Please run your program at least 5 times.
Round 1: Please choose three initial cluster centers as A1, A7, and A8
Round 2: Please choose three initial cluster centers as A2, A6, and A8
Round 3: Please choose three initial cluster centers as A3, A5, and A6
Round 4: Please choose three initial cluster centers as A2, A3, and A7
Round 5: Please randomly choose any three points from a two dimensional space (0,0) to (10,10) as initial three cluster centers.
Round 6 and More: Please randomly choose any three points from a two dimensional space (0,0) to (15,15) as initial three cluster centers.
Among all of these clustering results, please tell us which clustering result is the best one and why?
Submit your source code and a short note of how to execute it. Please turn in hard copy with the clustering results too.

2. (60 pt) The hierarchical agglomerative clustering algorithm is to cluster data from bottom to up. Here is the pseudo-code for it. Please implement hierarchical agglomerative clustering algorithm on 2-dimensional numerical data by C++, which should be fairly easy to derive from this.

```
SIMPLEHAC($d_1, . . . , d_N$)
    for $o \leftarrow 1$ to $N$
        for $p \leftarrow 1$ to $N$
            $C[o][p] \leftarrow$ SIM($d_o, d_p$)
        end
        $I[o] \leftarrow 1$ (keeps track of active clusters)
    end
    $A \leftarrow []$ (assembles clustering as a sequence of merges)
    for $k \leftarrow 1$ to $N - 1$
        $<i,m> \leftarrow \text{argmax}_{\{<i,m>:i \neq m \wedge I[i]=1 \wedge I[m]=1\}} C[i][m]$
        $A$.APPEND($<i,m>$) (store merge)
        for $j \leftarrow 1$ to $N$
```

```
            $C[i][j] \leftarrow$ SIM($i,m, j$)
            $C[j][i] \leftarrow$ SIM($i,m, j$)
        End
        $I$[m] $\leftarrow$ 0 (deactivate cluster)
    end
    return $A$
```

Use your code to cluster the following 12 points (with (x, y) representing locations) into clusters A1(2, 2) A2(3.01, 2) A3(4.02, 2) A4(5.03, 2) A5(6.04, 2) A6(7.05, 2) A7(2, 3.5) A8(3.01, 3.5), A9(4.02, 3.5), A10(5.03, 3.5), A11(6.04, 3.5) and A12 (7.05, 3.5). The distance (similarity) function between two points $a=(x1, y1)$ and $b=(x2, y2)$ is defined as L2-norm: $\rho(a, b) = ((x2 - x1)^2 + (y2 - y1)^2)^{1/2}$. Please use single linkage, complete linkage, and centroid linkage to generate three different dendrograms. In the code, SIM(i,m, j) needs to be altered based on the selected cluster distance metric (single linkage, complete linkage, and centroid). Based on the generated dendrogram, show the clustering result of **two** clusters and **six** clusters in single linkage, complete linkage, and centroid linkage.

Submit your source code and a short note of how to execute it. Submit hardcopy of dendrogram and clustering result too.