CSE 341: COMPUTER ORGANIZATION

Project (FALL 2019)

Milestone Due on **November 22, 2019**

Final Due on **December 6, 2019**

1. Project Description

In this project, you are required to design and implement the k-means clustering algorithm in assembly language. The implementation will be tested using the given testing data points.

K-Means is one of the most popular "clustering" algorithms, also the basic techniques in machine learning. K-means stores k centroids that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid. Given n sets of data points, k-means clustering is the iterative algorithm which tries to partition the data points into non-overlapping sets. You may refer to the Wikipedia for detailed explanation of the algorithm.
https://en.wikipedia.org/wiki/K-means_clustering

For the evaluation process, TA will implement your codes with unseen data points. You can check your codes running correctly with TA before submitting the report.

2. Submission Approach & Grading Criteria

1) Submission Approach

Submit your source file via UBlearns with name:

- *Report_Firstname_Lastname_UBpersonnumber.pdf*
- *K-means_Firstname_Lastname_UBpersonnumber.asm*
- *Euclidean_Firstname_Lastname_UBpersonnumber.asm*
- *Comparison_Firstname_Lastname_UBpersonnumber.asm*
- *Centroid_Firstname_Lastname_UBpersonnumber.asm*

2) Grading Criteria

- Euclidean Distance: **30%**
- Distance Comparison: **20%**
  (Bonus +10 points if the first two tasks are submitted before Milestone Due)

- Updating the centroids: **20%**
- System Integration and testing: **10%** (using the example 10 data points)
- **Report: 20%**

3) Early Grading Policy (Please read carefully!!!)

In order to incentivize and encourage the early completion of the project, I adopt an Early Grading policy for the course project.

If you have finished all of your codes (any time before the milestone/final deadlines), you can see TAs (or notify TAs that your submitted codes are ready to test) to check and verify your codes. If TAs' testing and verification are successful, you will be assured to get full credit for the coding part (80% points). But you still need to submit your codes and report on UBlearn.

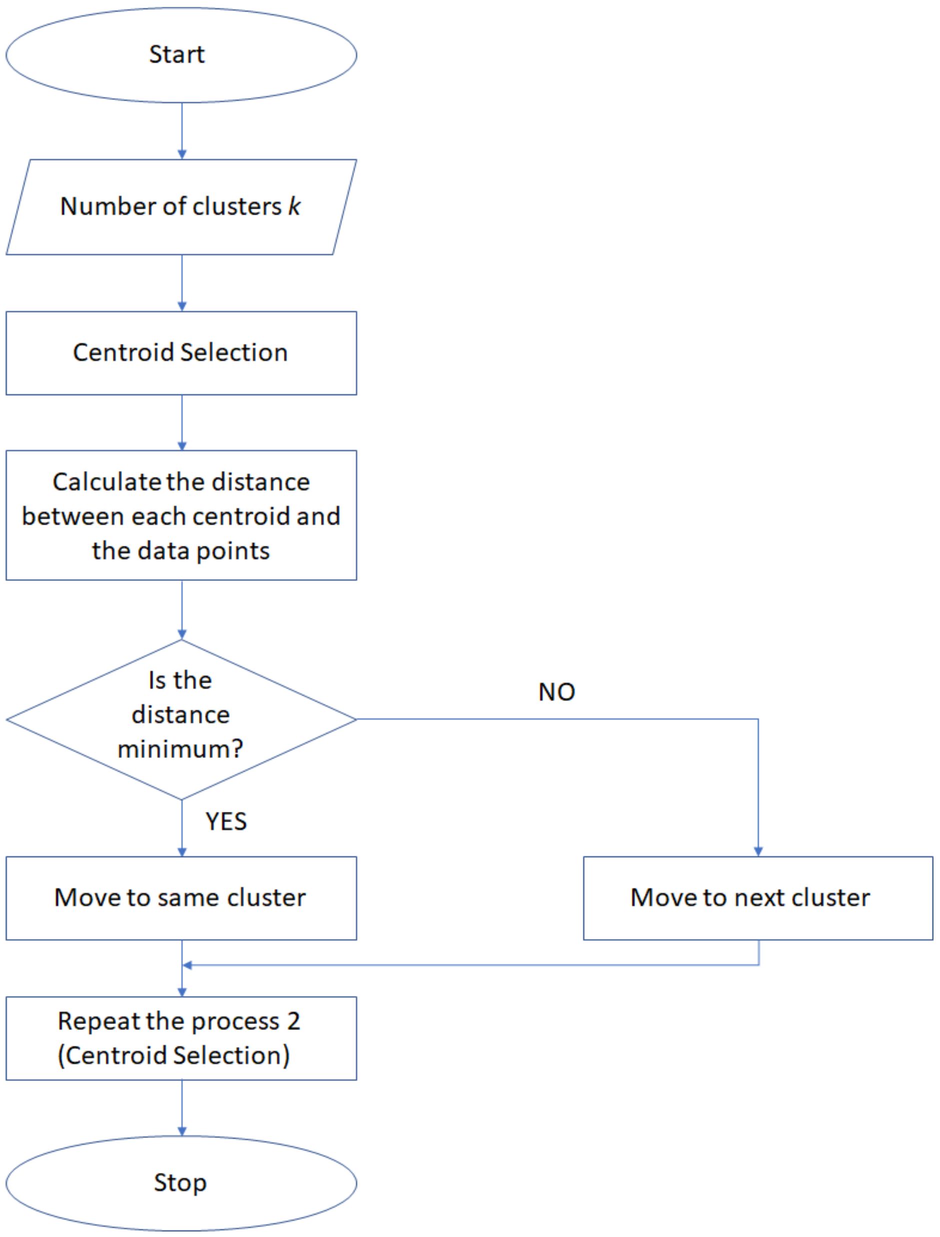
If the testing is not successful, you can go back to make debugging and corrections. However, any change will not be allowed after the final deadline.

4) Late Submission Policy

Submission after the final deadline (midnight on Dec. 6) will not be considered.

3. Flowchart & Algorithm Description

1) Flowchart



In your project, for the first iteration of centroid selection, you may randomly assign two points (out of 10 given points) as the initial centroids of two clusters respectively. Afterward, the centroid selection is calculating the center of mass of each cluster.

2) Algorithm Description

```
K-means($X = \{x_1, x_2, \dots, x_N\}, K$)
1     choose the initial centers $C = \{c_1, c_2, \dots, c_k\}$
2     for $k \leftarrow 1$ to $K$
3     do $\mu_k \leftarrow c_k$
4     while stopping condition has not met
5     do for $k \leftarrow 1$ to $K$
6         do $w_k \leftarrow \{\}$    ( cluster initialization)
7         for $n \leftarrow 1$ to $N$
8         do $j \leftarrow argmin_{i=1,\dots,k} ||\mu_k - x_n||^2$
9             $w_j \leftarrow w_j \cup \{x_n\}$   ( cluster updating )
10        for $k \leftarrow 1$ to $K$
11        do $\mu_k \leftarrow \frac{1}{|w_k|} \sum_{x \in w_k} x$    (centroid re-computation)
12    return $\{\mu_1, \mu_2, \dots, \mu_k\}$
```

3) Euclidean Distance

Euclidean distance defines the distance between two points in Euclidean space. When considering a one-dimension Euclidean space, the distance between A and B could be given by:

$$\sqrt{(A-B)^2} = |A-B|$$

For a two-dimension Euclidean space, if

$$A = (x_1, y_1), B = (x_2, y_2)$$

The Euclidean distance is given by:

$$\mathrm{Disc}(A,B) = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$$

4) Modified Euclidean Distance

In this project, for the convenience of integer calculations through the entire flow, you need to calculate the "modified" Euclidean distances between every two data points in a two-dimension Euclidean space, in order to properly cluster them into two groups. Basically, you don't need to calculate the square root.

$$\overline{\mathrm{Disc}(A,B)} = (x_2-x_1)^2 + (y_2-y_1)^2$$

Please note that, this is not the right way for K-means clustering. It is only simplified for the purpose of this project implementation.

4. C-code

As a reference, you may check the C codes implementing k-means clustering algorithm, developed by Ethan Brodsky at the University of Wisconsin – Madison.

http://homepages.cae.wisc.edu/~brodskye/mr/kmeans/

5. Example

Step 1: Let’s choose the initial center E(2,4) and I(6,2)

Step 2: Let’s calculate the distance between data points with each centroid

[Table 1]. Distance between Data point and centroids

	Centroid 1 E(2,4)	Centroid 2 I(6,2)
A (2,2)	4	16
B (5,3)	10	2
C (1,5)	2	34
D (3,3)	2	10
E (2,4)	0	20
F (2,1)	9	17
G (4,2)	8	4
H (5,1)	18	2
I (6,2)	20	0
J (5,2)	13	1

Step 3: Let’s cluster the data points based on their “modified Euclidean distances”.

Cluster 1 = { A, C, D, E, F }

Cluster 2 = { B, G, H, I, J }

Step 4: Let’s update the centroids.

(i) Center of the mass of the updated Cluster 1

X_coordinate = ((2+1+3+2+2))/5 = 10/5 = 2,

Y_coordinate = ((2+5+3+4+1))/5 = 15/5 = 3

(ii) Center of the mass of the updated Cluster 2

X_coordinate = ((5+4+5+6+5))/5 = 25/5 = 5

Y_coordinate = ((3+2+1+2+2))/5 = 10/5 = 2

-> Centroid 1: (2, 3)   Centroid 2: (5, 2)

Step 5：Repeat step 2.

[Table 1]. Distance between Data point and centroids		
	Centroid 1 (2, 3)	Centroid 2 (5, 2)
A (2,2)	1	9
B (5,3)	9	1
C (1,5)	5	25
D (3,3)	1	5
E (2,4)	1	13
F (2,1)	4	10
G (4,2)	5	1
H (5,1)	13	1
I (6,1)	20	2
J (5,2)	10	0

Step 6: Repeat step 3.

Cluster 1 = { A, C, D, E, F } -> Unchanged !

Cluster 2 = { B, G, H, I, J } -> Unchanged !

Step 7: Repeat step 4.

There is no change to the centroids and clusters. **Stopping condition is met.**
No need to repeat step 4.

Step 8: Finish