# CS2510 Project 3 Report

Zhuolun Li, Yi Yang

# Implementation

## WorkFlow

- MapReduce1
  - Map
    - Calculate cell id by data point's coordination
    - Generate KeyValue Pair <cellId, 1>
  - Reduce
    - Count the number of points in each cell
    - Generate KeyValue Pair (CellId, number of points).
- CellMerge
  - Get the result from MapReduce1 from hdfs and copy and merge to the local file system.
  - If the number of points in the Cell < k+1, merge the surrounding cells.
  - Create a HashMap Object to store the shape of current cells, which contains the margins of each cell.
  - Serialize the HashMap Object, and save to MapReduce's Configuration.
- MapReduce2
  - Map
    - Calculate cell id by data point's coordination and the merged cells' shape
    - Generate KeyValue Pair (CellId, (pointId, X, Y))
  - Reduce
    - Reduce KeyValue Pair (CellId, (pointId, X, Y)), add points within the same cell to the point's knnList.
    - Generate KeyValue Pair (PointId, knnList).
- MapReduce3
  - Map
    - Test if there are potential knn points in neighbor cells, and create String based on the result.
    - Generate KeyValue Pair (PotentialCellId, (PointId, X, Y, OriginalCellId, boolean)).
    - Generate KeyValue Pair (CellId, (PointId, X, Y)).
  - Reduce
    - Reduce the KeyValue Pair (PotentialCellId, (PointId, X, Y, OriginalCellId, boolean)) and (CellId, (PointId, X, Y))

        - Create a list of points in the cell.
        - Create a minimum heap, add potential points to the heap and get k points with the minimum distance.
        - Generate KeyValue Pair (PointId, (X, Y, CellId, knnList)).
- MapReduce4
    - Map
        - Generate KeyValue Pair (PointId, knnList).
    - Reduce
        - Reduce KeyValue Pair (CellId, (pointId, X, Y)), add points within the same cell to the point's knnList.
        - Generate KeyValue Pair (PointId, knnList).
- Save Result
    - Get the result from MapReduce1 from hdfs and copy and merge to the local file system.

# Experiment

## Data Set:

0,87,47
1,49,45
2,37,49
3,55,57
4,48,21
5,2,31
6,35,67
7,15,67
8,20,36
9,85,50
10,49,42
11,16,5
12,38,29
13,15,63
14,9,41
15,81,49
16,2,11
17,63,92
18,33,18
19,83,41
20,32,44
21,52,26

22,43,80
23,72,51
24,97,93
25,64,79
26,6,88
27,31,61
28,78,80
29,70,16
30,11,33
31,65,57
32,6,31
33,94,18
34,97,81
35,21,33
36,12,51
37,90,3
38,72,70
39,80,91
40,42,91
41,14,70
42,57,98
43,5,13
44,27,4
45,73,82
46,18,65
47,77,90
48,80,6
49,47,75

output:

0 [5=13.0, 43=15.264338]
1 [22=10.0, 10=10.29563]
2 [43=13.038404, 18=19.723083]
3 [20=3.6055512, 33=6.3245554]
4 [6=3.0, 47=11.7046995]
5 [0=13.0, 43=18.439089]
6 [4=3.0, 47=8.944272]
7 [8=9.219544, 31=12.369317]
8 [31=3.1622777, 12=4.1231055]
9 [19=5.656854, 24=6.708204]
10 [27=6.4031243, 1=10.29563]

11 [35=6.0, 44=6.4031243]
12 [31=1.0, 8=4.1231055]
13 [38=4.1231055, 15=7.81025]
14 [41=3.6055512, 17=8.5440035]
15 [18=2.236068, 13=7.81025]
16 [17=6.0827627, 41=12.0415945]
17 [41=6.0, 16=6.0827627]
18 [15=2.236068, 13=10.0]
19 [9=5.656854, 24=7.28011]
20 [33=3.0, 3=3.6055512]
21 [20=8.246211, 33=9.433981]
22 [1=10.0, 27=12.0415945]
23 [46=8.944272, 24=13.928389]
24 [9=6.708204, 19=7.28011]
25 [49=5.656854, 40=17.117243]
26 [42=7.071068, 9=8.5440035]
27 [33=6.0, 10=6.4031243]
28 [22=12.727922, 1=17.262676]
29 [37=5.0990195, 32=6.708204]
30 [7=12.649111, 40=12.727922]
31 [12=1.0, 8=3.1622777]
32 [37=5.3851647, 36=6.0827627]
33 [20=3.0, 27=6.0]
34 [46=13.152946, 39=15.811388]
35 [44=5.3851647, 11=6.0]
36 [32=6.0827627, 29=7.615773]
37 [29=5.0990195, 32=5.3851647]
38 [13=4.1231055, 26=9.0]
39 [47=9.848858, 6=13.0]
40 [30=12.727922, 49=13.152946]
41 [14=3.6055512, 17=6.0]
42 [26=7.071068, 8=10.440307]
43 [2=13.038404, 0=15.264338]
44 [35=5.3851647, 11=6.4031243]
45 [34=19.924858, 5=25.0]
46 [23=8.944272, 34=13.152946]
47 [6=8.944272, 39=9.848858]
48 [35=9.219544, 44=13.038404]
49 [25=5.656854, 40=13.152946]