Assignment 3

1. Matrix Multiplication

Let's take an example if we have two matrices. **Matrix A** is **2 X 5. Matrix B** is **5 X 3**

Matrix A

0	1	2	3	4
5	6	7	8	9

Matrix B

0	1	2
3	4	5
6	7	8
9	10	11
12	13	14

Let's call resultant matrix as **Matrix C**. Matrix C will be of dimension **2 X 3**.
In matrix multiplication, 1st block of matrix C i.e., (0,0) will be computed as sum of ith row of matrix A * column of matrix B. Below is the formula:

$$(\mathbf{AB})_{ij} = \sum_{k=1}^{m} A_{ik} B_{kj} \, .$$

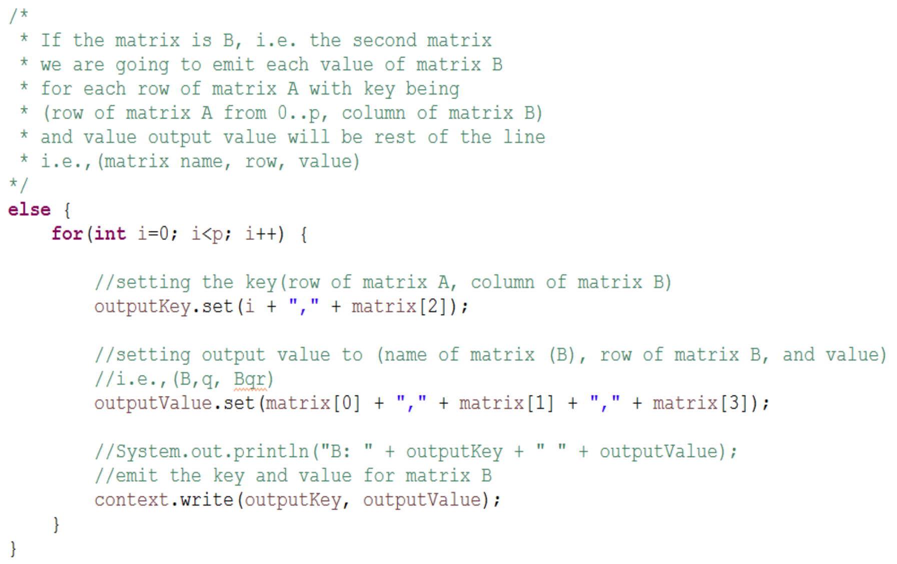
For matrix multiplication in map reduce, we need to create (key, value) pairs in map and later do the multiplication and addition in the reduce. By creating key, value pair, we can do computation in parallel and large matrices can be multiplied in the less time.

In the Map, we concentrate on creating (key, value) pair. First we get the number of **rows in Matrix A (i.e., p)** and **columns in Matrix B (i.e., r)**. For the parallel computation, we need to emit the values multiple times with different keys, number of emission of each value depend upon the number of times are they are going to be used in computation. Since the number at (0,0) i.e., 0th row and 0th column in Matrix A is going to multiplied by the first value of every column of B. Therefore, number of time the (0,0) value of matrix A emitted is number of columns in matrix B. Similarly, for matrix B, the value at (0,0) emitted is number of rows in matrix A. Since the values are emitted multiple times the key would be composite.
Key for Matrix A is (Row Number of Matrix A, Particular column of Matrix B). In the above example, the value 1 is emitted with key (1,0) where 1 is the row of matrix A and 0 is the column of matrix B. Value 1 will be emitted thrice with keys (1,0), (1,1) and (1,2). **Value for Matrix A** is (matrix name, column number, value), for value 1, the value would be (A, 1, 1).

```
//checking for the matrix name in the line
if(matrix[0].equals("A")) {

    /*
     * If the matrix is A, i.e. the first matrix
     * we are going to emit each value of matrix A
     * for each column of matrix B with key being
     * (row of matrix A, column number from 0..r)
     * and value output value will be rest of the line
     * i.e.,(matrix name, column, value)
    */
    for(int i=0; i<r; i++) {

        //setting output key to (row of matrix A, column of final matrix C)
        //i.e., (p,r)
        outputKey.set(matrix[1] + "," + i);

        //setting output value to (name of matrix (A), column of matrix A, and value)
        //i.e., (A, q, Apq)
        outputValue.set(matrix[0] + "," + matrix[2] + "," + matrix[3]);

        //System.out.println("A: " + outputKey + " " + outputValue);
        //emit the key and value for matrix A
        context.write(outputKey, outputValue);
    }
}
```

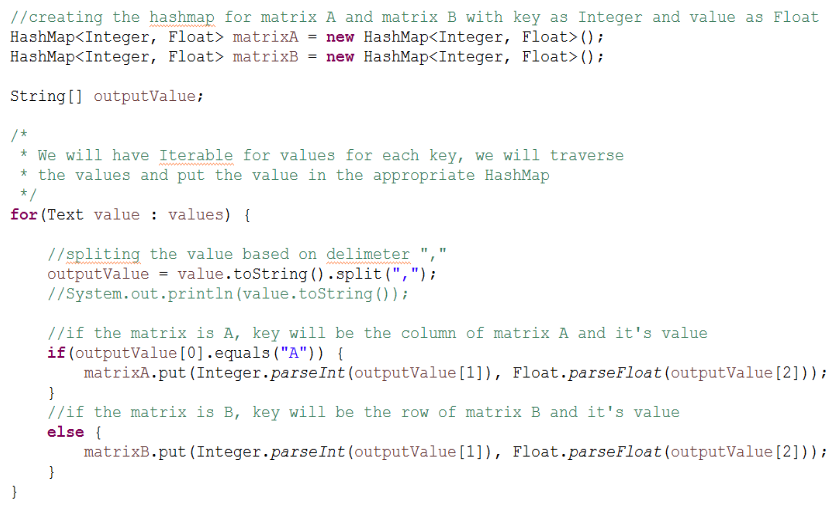
Assignment 3

Key for Matrix B is (particular row of Matrix A, column of Matrix B). In the above example, the value 3 is emitted with key (0,0) where 0 is the row of matrix A and 0 is the column of matrix B. Value 3 is emitted twice with key (0,0) and (0,1). **Value for Matrix B** is (matrix name, row number, value), for value 3, the value would be (B, 1, 3).



```
/*
 * If the matrix is B, i.e. the second matrix
 * we are going to emit each value of matrix B
 * for each row of matrix A with key being
 * (row of matrix A from 0..p, column of matrix B)
 * and value output value will be rest of the line
 * i.e.,(matrix name, row, value)
*/
else {
    for(int i=0; i<p; i++) {

        //setting the key(row of matrix A, column of matrix B)
        outputKey.set(i + "," + matrix[2]);

        //setting output value to (name of matrix (B), row of matrix B, and value)
        //i.e.,(B,q, Bqr)
        outputValue.set(matrix[0] + "," + matrix[1] + "," + matrix[3]);

        //System.out.println("B: " + outputKey + " " + outputValue);
        //emit the key and value for matrix B
        context.write(outputKey, outputValue);
    }
}
```

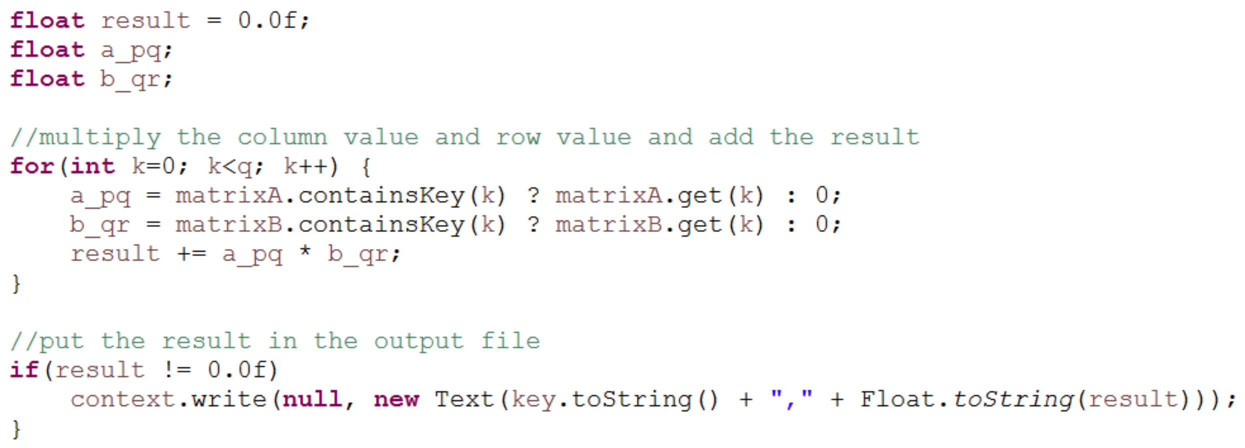
In the Reducer, every reducer will get all the values of a particular key. First we get the **number of columns in matrix A or number of rows in matrix B (i.e, q).** Since every block of the resultant matrix is the addition of multiplication of the value of ith row of matrix A and value of ith column of matrix B where i is from 0 to q. For the computation, we create two HashMap one for each matrix. In the HashMap, column number for matrix A becomes the key and row number for matrix B becomes the key. Later we traverse through the hashmap from 0 to q and the get the values from hashmap for each traversal and multiply and add to the result.



```
//creating the hashmap for matrix A and matrix B with key as Integer and value as Float
HashMap<Integer, Float> matrixA = new HashMap<Integer, Float>();
HashMap<Integer, Float> matrixB = new HashMap<Integer, Float>();

String[] outputValue;

/*
 * We will have Iterable for values for each key, we will traverse
 * the values and put the value in the appropriate HashMap
 */
for(Text value : values) {

    //spliting the value based on delimeter ","
    outputValue = value.toString().split(",");
    //System.out.println(value.toString());

    //if the matrix is A, key will be the column of matrix A and it's value
    if(outputValue[0].equals("A")) {
        matrixA.put(Integer.parseInt(outputValue[1]), Float.parseFloat(outputValue[2]));
    }
    //if the matrix is B, key will be the row of matrix B and it's value
    else {
        matrixB.put(Integer.parseInt(outputValue[1]), Float.parseFloat(outputValue[2]));
    }
}
```

Creating the HashMap

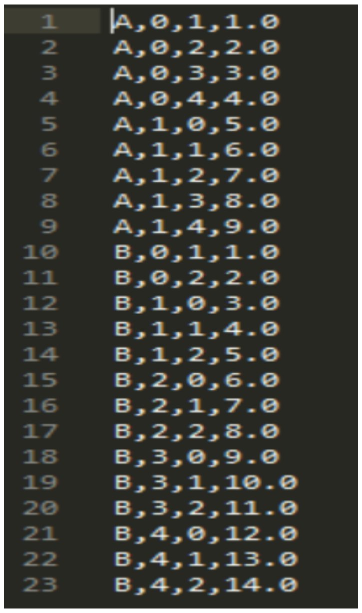
Assignment 3



```java
float result = 0.0f;
float a_pq;
float b_qr;

//multiply the column value and row value and add the result
for(int k=0; k<q; k++) {
    a_pq = matrixA.containsKey(k) ? matrixA.get(k) : 0;
    b_qr = matrixB.containsKey(k) ? matrixB.get(k) : 0;
    result += a_pq * b_qr;
}

//put the result in the output file
if(result != 0.0f)
    context.write(null, new Text(key.toString() + "," + Float.toString(result)));
}
```

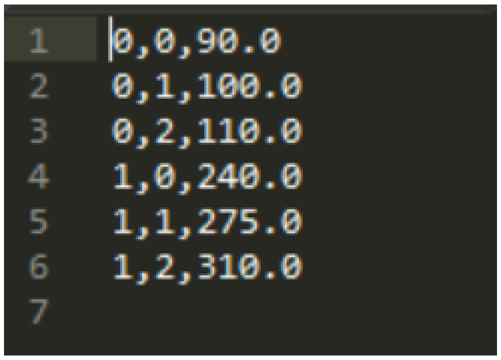
Computing the Result

Input



```
A,0,1,1.0
A,0,2,2.0
A,0,3,3.0
A,0,4,4.0
A,1,0,5.0
A,1,1,6.0
A,1,2,7.0
A,1,3,8.0
A,1,4,9.0
B,0,1,1.0
B,0,2,2.0
B,1,0,3.0
B,1,1,4.0
B,1,2,5.0
B,2,0,6.0
B,2,1,7.0
B,2,2,8.0
B,3,0,9.0
B,3,1,10.0
B,3,2,11.0
B,4,0,12.0
B,4,1,13.0
B,4,2,14.0
```

Output

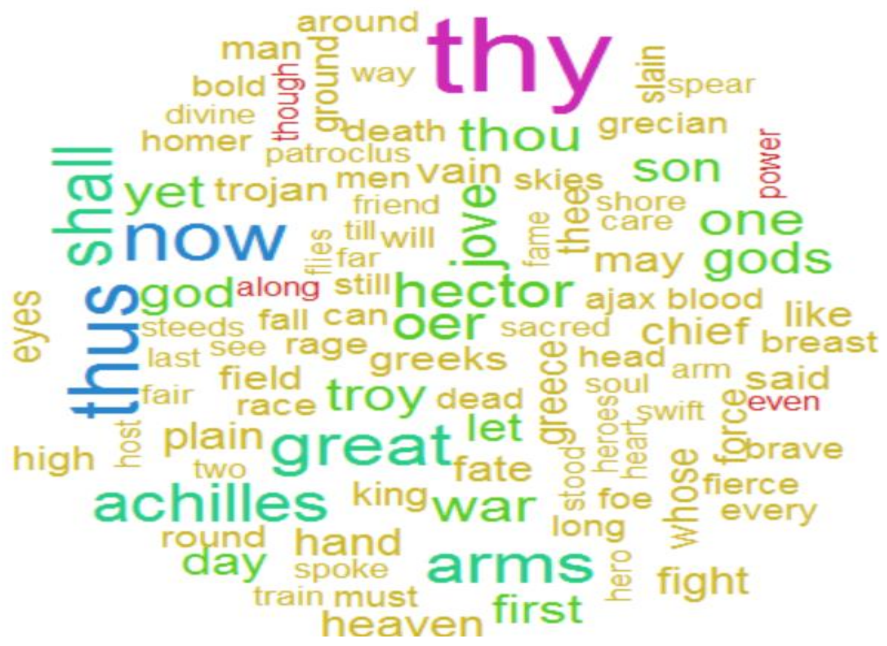


```
0,0,90.0
0,1,100.0
0,2,110.0
1,0,240.0
1,1,275.0
1,2,310.0
```

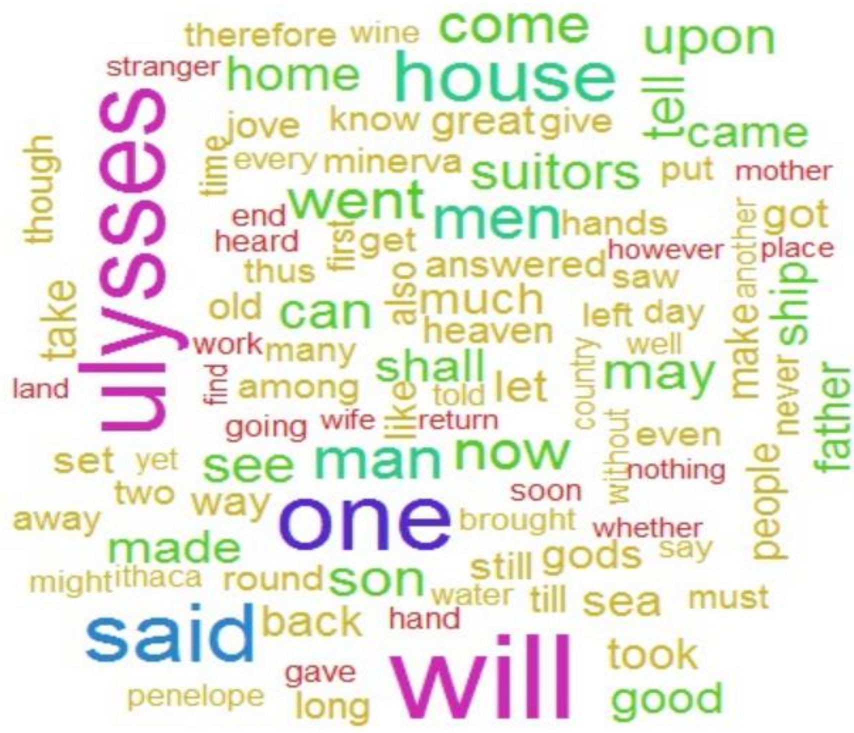
Assignment 3

2. <u>Text Mining in R – Word Cloud</u>

a. Word Cloud – Iliad



b. Word Cloud – Odyssey

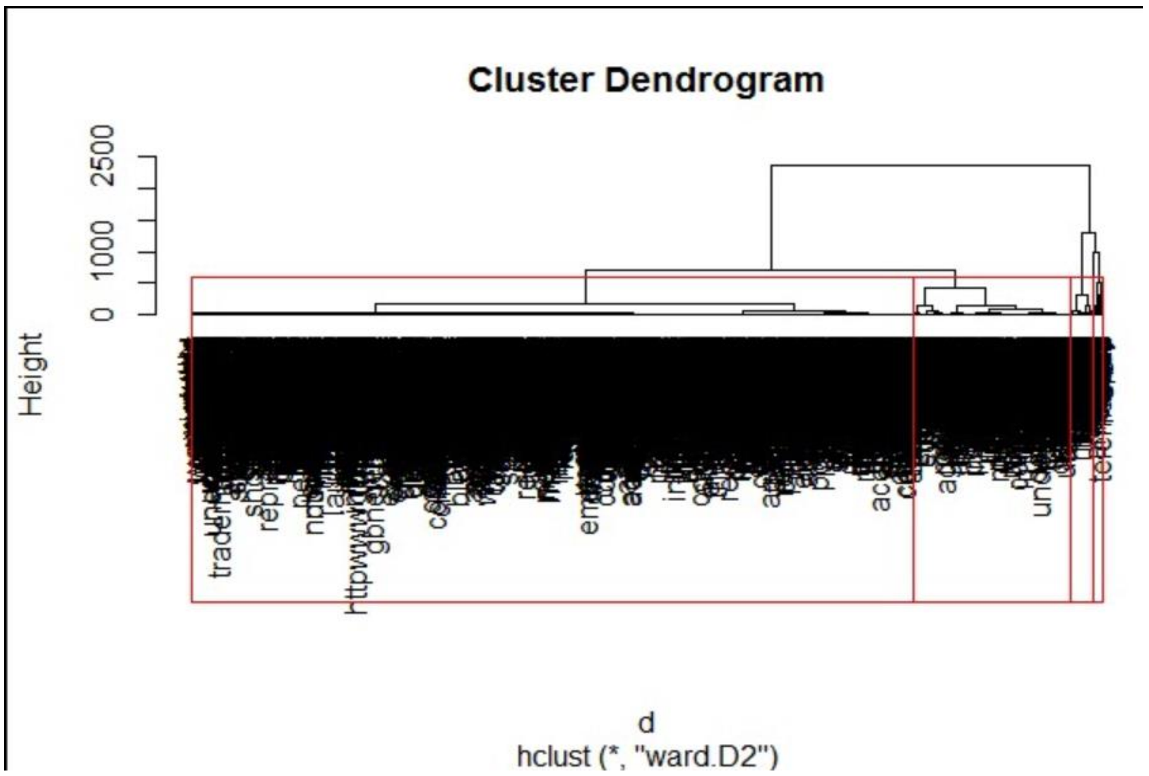


Assignment 3

c. Clustering

1. Hierarchal Clustering

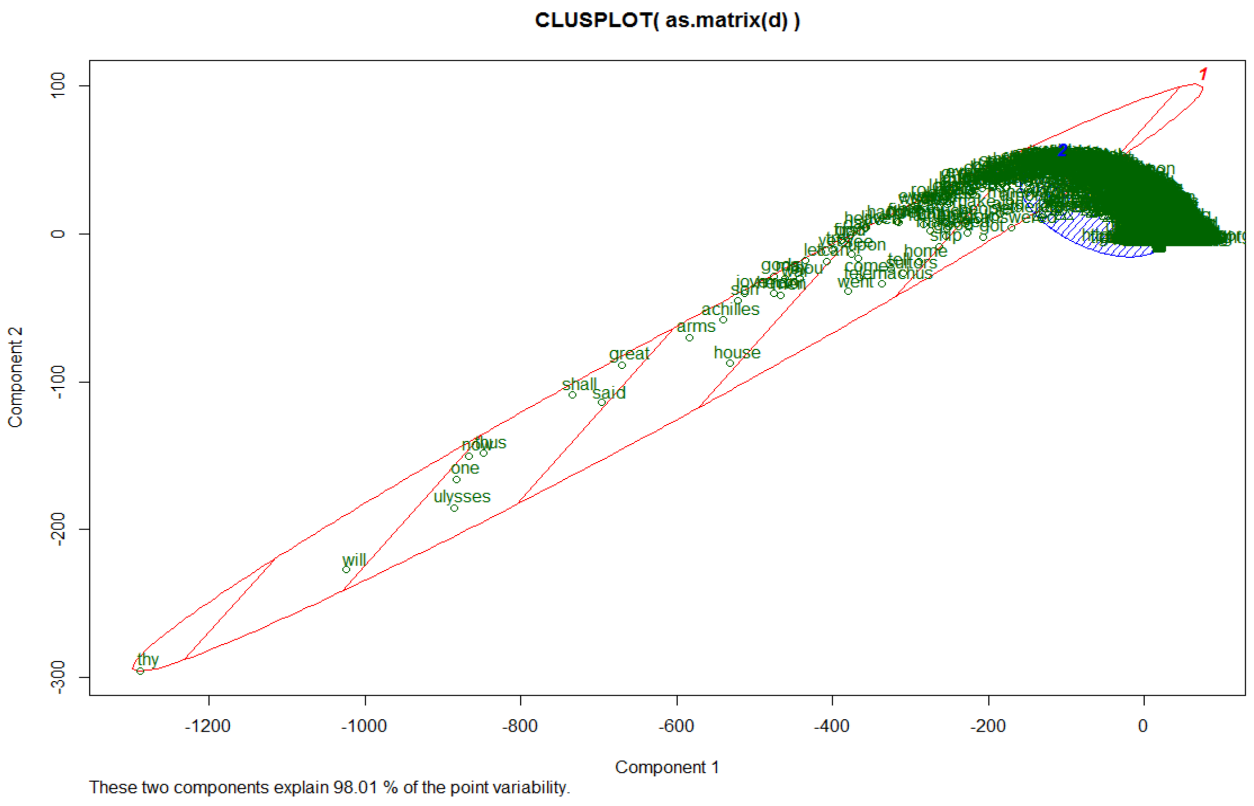
Cluster will group the **Term by similarity**. In the below **Hierarchal Clustering**, we are making **5 clusters** with the distance type as "**Euclidian**". You can choose any number of cluster and distance type can be "**Manhattan**" as well. Hierarchal Clustering will create **Cluster Dendrogram**, the red line are the borders for each group, which will group the words which comes under same hierarchy. In the below cluster, there are 4653 words and only 5 clusters, we can create more clusters for this large number for words.



2. K-Means Clustering

In the below image, we have created two clusters one with red eclipse and one with blue eclipse. The amount of variance explained is related to the two principal components calculated to visualize your data. In the char, words like Ulysses, one, thy clearly fall in the cluster one i.e., red. But the words which are in between red and blue will be hard to predict, the prediction by the algorithm will be mostly based on the Euclidean distance.

Assignment 3



d. Characteristics of Iliad and Odyssey

1. Iliad and Odyssey are texts written in Greek.
2. **In Odyssey**, Greek hero Ulysses is referred lot of times whereas **In Iliad**, Achilles is referred.
3. **In Odyssey**, there are words like home, come, mother, son. It is more related to the family or person coming to home.
4. **In Iliad**, there are words like war, death, arm, rage, fall. It seems like it is related to war or anger.
5. There are not many things common in the text.