CS585: Big Data Management

Project 2

"Choose 2 out of the 3 questions"

Total Points: **70**

Release Date: **06/15/2022**

Due Date: **06/25/2022 (11:59PM)**

Short Description

In this project, you will write java map-reduce jobs that implement advanced operations in Hadoop.

Problem 1 (Spatial Join) [35 points]

Spatial join is a common type of joins in many applications that manage multi-dimensional data. A typical example of spatial join is to have two datasets: ***Dataset P*** (set of points in two-dimensional space) as shown in Figure 1a, and ***Dataset R*** (set of rectangles in two-dimensional space) as shown in Figure 1b. The spatial join operation is to join these two datasets and report any pair (rectangle *r*, point *p*) where *p* is contained inside *r* (or on the boundaries of *r*).

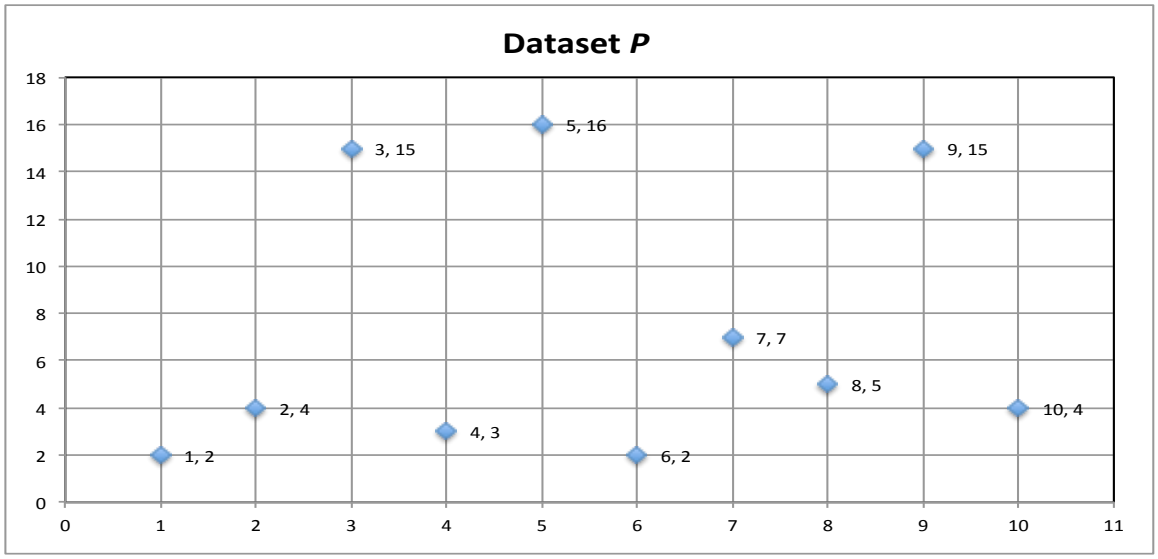


Figure 1a: Set of 2D Points

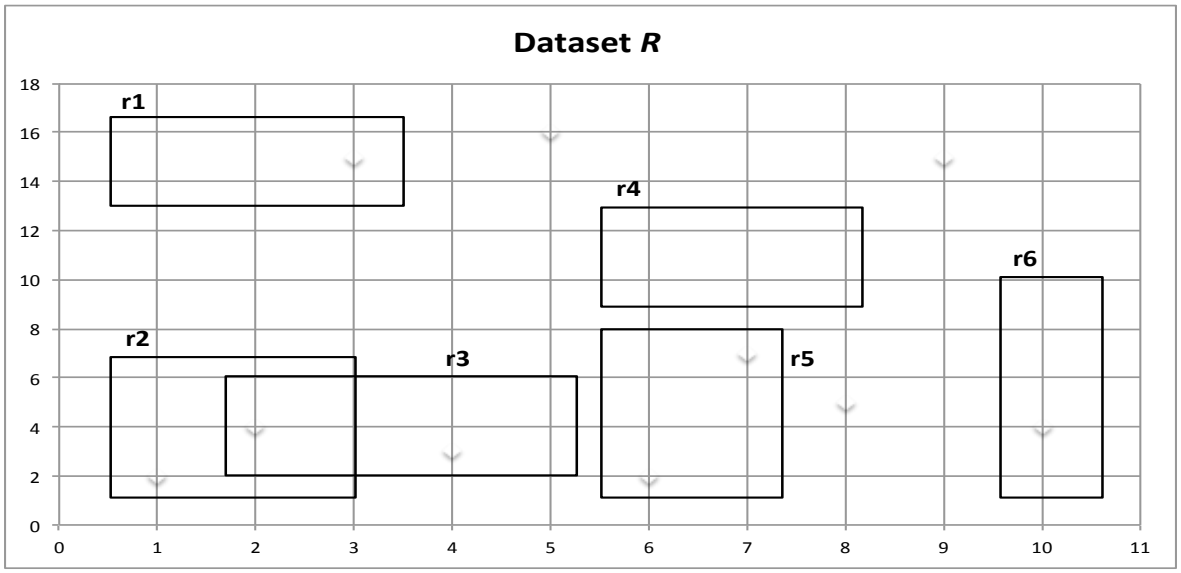


Figure 1b: Set of 2D Rectangles

For example, the join between the two datasets shown in Figure 1, will result in.

<r1, (3,15)>
<r2, (1,2)>
<r2, (2,4)>
<r3, (2,4)>
<r3, (4,3)>
<r5, (6,2)>
<r5, (7,7)>
<r6, (10,4)>

Step 1 (Create the Datasets)[10 Points]

- Your task in this step is to create the two datasets ***P*** (set of 2D points) and ***R*** (set of 2D rectangles). Assume the space extends from 1…10,000 in both the *x* and *y* axis. Each line in file P should contain one point, and each line in file R should contain one rectangle.
- Scale each dataset *P* and *R* to be at least 100MB.
- Choose the appropriate random function (of your choice) to create the points. When the data is created it must be random with no specific order.
- For rectangles, select one point at random and consider this point as *the bottom-left corner*, and then select two random variables that define the *height* and *width* of the rectangle. For example, the height random variable can be uniform between [1,20] and the width is also uniform between [1,7]. Therefore, a rectangle is defined as <*bottomLeft_x*, *bottomleft_y*, *h*, *w*>.
- You can assume all coordinates are integer values.

Step 2 (MapReduce job for Spatial Join)[25 Points]

In this step, you need to write a java map-reduce job that implements the spatial join operation between the two datasets *P* and *R* based on the following requirements:

- The program takes an optional input parameter W(x1, y1, x2, y2) that indicate a spatial window (rectangle) of interest within which we want to report the joined objects. If W is given, then any rectangle that is entirely outside W and any point that is outside W should be skipped. If W is omitted, then the entire two sets should be joined.
    - Example, referring to Figure 1, if the window parameter is W(1, 3, 3, 20), then the reported joined objects should be:

      <r1, (3,15)>
      <r2, (2,4)>
      <r3, (2,4)>

- You should have a single map-reduce job (many mappers and many reducers but in a single job) to implement the spatial join operation.

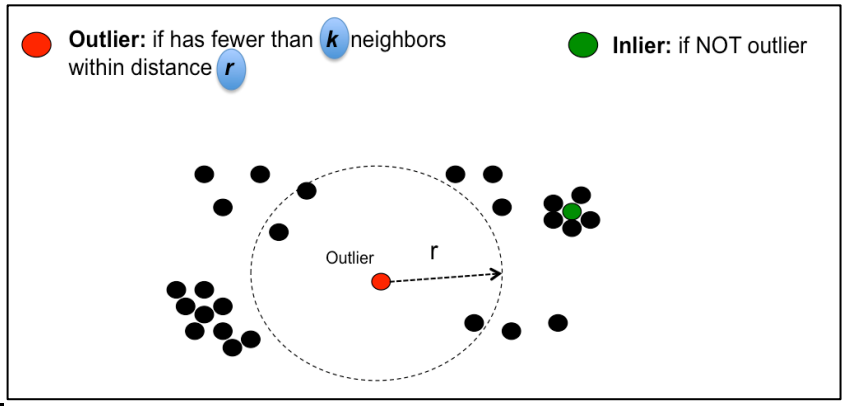
Problem 2 (Distance-Based Outlier Detection Clustering) [35 points]

Outliers are objects in the data that do not conform to the common behavior of the other objects. There are many definitions for outliers. One common definition is "*distance-based outliers*". In this definition (see the figure below), you are given two parameters, radius *r* and threshold *k*, and a point ***p*** is said to be outlier iff:

"Within a circle around p (p is the center) of radius r, less than k neighbors are found"

And point p is said to be inlier (Not outlier) iff:

"Within a circle around p (p is the center) of radius r, more than or equal to k neighbors are found"



Step 1: Dataset

Use the dataset P that you created in Problem 1. And you know the entire space boundaries, i.e., from (1,1) to (10,000, 10,000). The initial points in the HDFS file are totally in random order, and there is no specific organization. For example, referring to Figure 1(a), points (3,15), (10,4), and (4,3) can be in the 1st HDFS block, etc.

Step 2: Reporting Outliers (35 Points)

In this step, you need to write a java map-reduce job that ***reports the outliers*** based on the following requirements:

(1) The program takes two mandatory parameters *r* and *k*. If either is missing, then report an error.
(2) You must use a single map-reduce job (many mappers and many reducers but in a single job) to complete the task.
   a. If used more than one job, then for each extra job you will lose 15 points.
   b. If you assume single map and single reduce, then you will lose 25 points

Hint: Think of diving the space in small segments. Try to make the processing of each segment independent from any other segment. That is, for a specific point p, you should be able to decide whether it is outlier or not only based on the points in p's segment.

What to Submit (for each student)

- You will submit a single zip file containing all problems *{Java programs for Creating Data Files, Java code for the MapReduce Queries }*.
- For the java code, you need to submit the source code
- The zip file should also include a "Readme.pdf" file. In this file include:
    - Any assumptions that you have made
    - Create a table as shown below and fill it up

Question	Status (Select one) Fully Working/ Partially Working/ Not Working	Comment
Q1		
Q2		

    - Any comments you would like to provide regarding your code.

How to Submit

- Use the Canvas system to submit your files.