Ira A. Fulton Schools of Engineering
Arizona State University

CSE 511: Data Processing at Scale

Hot Spot Analysis Project

Purpose

In this project, you will perform numerous spatial queries on a huge database that contains geographic information and the real-time locations of the customers of a well-known taxi company.

Objectives

Learners will be able to:

- Set up Apache Spark and use dataframes within it to manage data operations.
- Write some simple Scala code then identify how to run it.
- Interact with geospatial data and run queries on it.

Technology Requirements

- Apache Spark 3.4.1
- SparkSQL 3.4.1
- Scala 2.13
- Java 17.0.8
- Hadoop 3.3.6
- Python 3.10
- SBT 1.9.4

Project Description

You will be utilizing Scala and Apache Spark in this project to create a solution that can extract crucial data from the provided dataset, which can then be used to make operational and strategic choices. The technology gives the client access to statistically significant geographic locations, which it can utilize to plan its operations in advance and improve customer service.

Please review the **Additional Resources: Hot Spot Analysis Project** before beginning. This is located in your course *Welcome and Start Here* module.

Note: Project details in the Overview Document may have been updated since the recording of the videos, so some directions or items may not match perfectly. Please follow the Overview Document's directions to complete your work correctly.

Directions

Before getting started, download the following project files from the course

1. CSE 511_Hot Spot Analysis Project_Required Templates.zip
2. CSE 511_Hot Spot Analysis Project_yellow_trip_sample_100000.zip

In this project, you are required to do spatial hot spot analysis. In particular, you need to complete two different hot spot analysis tasks.

Tasks

1. Hot Zone Analysis

This task will need to perform a range join operation on a rectangle datasets and a point dataset. For each rectangle, the number of points located within the rectangle will be obtained. The hotter rectangle means that it includes more points. So this task is to calculate the hotness of all the rectangles.

2. Hot Cell Analysis

This task will focus on applying spatial statistics to spatio-temporal big data in order to identify statistically significant spatial hot spots using Apache Spark. The topic of this task is from ACM SIGSPATIAL GISCUP 2016.

- Problem Definition page
- Submit Format page
- Special requirement (different from GIS CUP)

As stated in the Problem Definition page, in this task, you are asked to implement a Spark program to calculate the Getis-Ord statistic of NYC Taxi Trip datasets. We call it "**hot cell analysis**"

To reduce the computation power need, we made the following changes:

1. The input will be "yellow_trip_sample_100000.csv" dataset.
2. Each cell unit size is 0.01 * 0.01 in terms of latitude and longitude degrees.
3. You only need to consider Pick-up Location.

4. We don't use Jaccard similarity to check your answer. However, you don't need to worry about how to decide the cell coordinates because the code template generates cell coordinates. You just need to write the rest of the task.

Coding Template Specification

Input Parameters

1. Output path (Mandatory)
2. Task name: "hotzoneanalysis" or "hotcellanalysis"
3. Task parameters: (1) Hot zone (2 parameters): nyc taxi data path, zone path(2) Hot cell (1 parameter): nyc taxi data path

Example

< / > Generic code block | Make it interactive

```
1  test/output hotzoneanalysis src/resources/point-hotzone.csv
2  src/resources/zone-hotzone.csv hotcellanalysis
3  src/resources/yellow_trip_sample_100000.csv
```

Note

1. The number/order of tasks do not matter.
2. But, the first seven (7) of our final test cases will be hot zone analysis, the last eight (8) will be hot cell analysis.

Input Data Format

The main function/entrance is "cse512.Entrance" scala file.

1. **Point data**: The input point dataset is the pickup point of New York Taxi trip datasets. The data format of this phase is the original format of NYC taxi trip which is similar from Phase 2. But the coding template already parsed it for you. Find the data in the **.zip file** (attached in the Project Overviews and Resoures page, titled “...Yellow Trip Sample 100000”).
2. **Zone data** (only for hot zone analysis): at "src/resources/zone-hotzone" of the template

Hot Zone Analysis

The input point data can be any small subset of the NYC taxi dataset.

Hot Cell Analysis

The input point data is a sample dataset like "yellow_trip_sample_100000.csv."

Output Data Format

Hot Zone Analysis:

All zones with their count, sorted by "rectangle" string in an ascending order.

```
< / >  Generic code block    Make it interactive
1  "-73.795658,40.743334,-73.753772,40.779114",1
2  "-73.797297,40.738291,-73.775740,40.770411",1
3  "-73.832707,40.620010,-73.746541,40.665414",20
```

Hot Cell Analysis

The coordinates of top 50 hottest cells sorted by their G score in a descending order.

- Note: **Do not output** G score.

```
< / >  Generic code block    Make it interactive
1  -7399,4075,15
2  -7399,4075,29
3  -7399,4075,22
```

Example answers

An example input and answer are put in "testcase" folder of the coding template

Where you need to make changes:

Do not delete any existing code in the coding template unless you see this "**You need to change this part.**"

Hot Zone Analysis

In the code template:

1. You need to change "**HotzoneAnalysis.scala**" and "**HotzoneUtils.scala**".
2. The coding template has loaded the data and written the first step, range join query, for you. Please finish the rest of the task.
3. The output DataFrame should be sorted by you according to the "rectangle" string.

Hot Cell Analysis

In the code template,

1. You need to change "**HotcellAnalysis.scala**" and "**HotcellUtils.scala**".
2. The coding template has loaded the data and decided the cell coordinate, x, y, z and their min and max. Please finish the rest of the task.
3. The output DataFrame should be sorted by you according to G-score. The coding template will take the first 50 to output. **Do not output** G-score.

Report

In addition to your ipynb file, write a **2-3 page report detailing your work on the project**. Your report should be a PDF titled with "Last Name_First Name_CSE511_Hot Spot Analysis Project Report." Your report should include:

1. **Reflection**: How did you approach the project? What did you specifically do?
2. **Lessons Learned**: What did you learn by doing this project?
3. **Implementation**: You need to include the following parts in your project report:
    a. What hot zone analysis is doing:
        i. def ST_Contains(queryRectangle: String, pointString: String)
    b. What hot cell analysis is doing:
        i. write the multiple steps required before calculating the z-score.

Submission Directions for Project Deliverables

You must submit each of your NoSQL Project deliverables through Gradescope and Canvas. Carefully review submission directions outlined in this overview document in order to correctly earn

credit for your work. Learners may not email or use other means to submit any assignment or project for review, including feedback, and grading.

The NoSQL Project includes two (2) deliverables:

- **JAR file:** Title your file as “submission.jar” and submit it to Gradescope.
- **Report**: Submit your PDF report to Gradescope titled “Last Name_First Name_CSE511_Hot Spot Analysis Project Report”.

Gradescope Submission

Your submission will be reviewed by the course team and then, after the due date has passed, your score will be populated from Gradescope into your grade. You will receive credit for the report if you covered all the mentioned parts.

1. Go to the Canvas Assignment, “**Submission: Hot Spot Analysis Project**”.
2. Click the "**Load Submission…in new window**" button.
3. Once in Gradescope, select the project titled "**Hot Spot Analysis Project**" and a pop-up window will appear.
4. In the pop-up window, submit your “**submission.jar**” file and the **PDF report** for grading.
5. If needed: to resubmit the project in Gradescope:
    a. Return to the Canvas submission and open Gradescope.
    b. You will be navigated to the "Autograder Results" page (if it is not your first submission).
    c. Click the "**Resubmit**" button on the bottom right corner of the page and repeat the process from Step 3
6. You need to make sure your code can compile and package by entering sbt clean assembly. We will run the compiled package on our cluster directly using "spark-submit" with parameters. If your code cannot compile and package, you will not receive any points.

****Important**: Grading will take about 10~20 minutes.**

Tips (Optional)

This section is the same as that in Phase 2.

How to debug your code in IDE:

If you are using the Scala template,

1. Use IntelliJ Idea with Scala plug-in or any other Scala IDE.
2. Replace the logic of User Defined Functions ST_Contains and ST_Within in SpatialQuery.scala.
3. Append .master("local[*]") after .config("spark.some.config.option", "some-value") to tell IDE the master IP is localhost.
4. In some cases, you may need to go to "build.sbt" file and change % "provided" to % "compile" in order to debug your code in IDE
5. Run your code in IDE.
6. **You must revert Step 3 and 4 above and recompile your code before use of spark-submit!**

How to submit your code to Spark

If you are using the Scala template

1. Go to the project root folder.
2. Run sbt clean assembly. You may need to install sbt in order to run this command.
3. Find the packaged jar in "./target/scala-2.13/CSE511-Project-Hotspot-Analysis-Template-assembly-0.1.0.jar"
4. Submit the jar to Spark using Spark command "./bin/spark-submit". A pseudo code example: ./bin/spark-submit ~/GitHub/CSE511-Project-Hotspot-Analysis-Template/target/scala-2.13/CSE511-Project-Hotspot-Analysis-Template-assembly-0.1.0.jar test/output hotzoneanalysis src/resources/point-hotzone.csv src/resources/zone-hotzone.csv hotcellanalysis src/resources/yellow_trip_sample_100000.csv

Evaluation

There are 15 test cases for a total of 15 points. The first seven (7) of the final test cases will be hot zone analysis, the last eight (8) will be hot cell analysis. **If the submission fails, you will see the corresponding error logs that indicate where the error occurred.**

Grading Rubric for Report

Rubrics communicate specific criteria for evaluation. Prior to starting any graded coursework, learners are expected to read through the rubric so they know how they will be assessed. You are encouraged to self-assess your responses and make informed revisions before submitting your final report. Engaging in this learning practice will support you in developing your best work.

Component	0	1	2
Reflection	There is no reflection included.	The reflection attempts to demonstrate thinking about learning but is vague and/or unclear about the personal learning process.	The reflection explains the student's own thinking and learning processes, as well as implications for future learning.
Analysis	There is no analysis included.	The reflection attempts to analyze the learning experience but the value of the learning to the student or others is vague and/or unclear.	The reflection is an in-depth analysis of the learning experience, the value of the derived learning to self or others, and the enhancement of the student's appreciation for the discipline.

Learner Checklist

Prior to submitting, read through the Learner Checklist to ensure you are ready to submit your best work.

- ☐ Did you title your file correctly and convert it into a single **.jar file**?
- ☐ Did you include your legal first and last name in the designated area on the **report**?
- ☐ Did you title your **report document** correctly and convert it into a single **pdf**?
    - **Last Name_First Name_CSE###_Name of Project**
- ☐ Did you answer all of the questions to the best of your ability?
- ☐ Did you make sure your answers directly address the prompt(s) in an organized manner that is easy to follow?
- ☐ Did you self-assess your open-ended responses using the rubric and make any necessary revisions?
- ☐ Did you proofread your work?