Building and Running the Google Web Graph App with Spark and SBT

This document outlines the steps to build and run the `GoogleWebGraphApp` application using Spark and SBT. The application uses Apache Spark's GraphX library to load the Google Web Graph, calculate PageRank scores, and print the results.

Prerequisites

1. **Apache Spark**: Ensure you have a Spark cluster running with a master and worker nodes (using Docker in this case).
2. **SBT (Scala Build Tool)**: Used to compile the project and build a JAR file.
3. **Docker**: For running Spark cluster and building the project.
4. **Docker Compose**: For setting up Spark master and worker nodes.

Project Structure

The project directory contains the following structure:

```
/graphx-scripts
      Dockerfile (optional, if needed for customization)
      sbt-config.sbt
      src
          main
              scala
                  GoogleWebGraphApp.scala
      target
          scala-2.12
              graphx-scripts_2.12-0.1.jar (Generated JAR after building)
```

Step 1: Build the Project Using SBT

In order to build the project using SBT, the following steps need to be executed within the Docker container running the `sbt-builder` service.

1. **Navigate to the project directory** (inside the Docker container).

   If using Docker Compose, the `sbt-builder` container mounts the project directory at `/opt/graphx-scripts`. Use this path to access the files.

2. **Run the SBT command to build the project**:

   The SBT command will compile and package the Scala application into a JAR file.

   ```
   sbt package
   ```

This will create a JAR file in the `target/scala-2.12/` directory. The file will be named `graphx-scripts_2.12-0.1.jar`.

Step 2: Submit the Job to Spark Using `spark-submit`

Once the JAR file is built, the job can be submitted to the Spark cluster using the `spark-submit` command. This step is executed from your local machine or the Docker container that has access to the Spark cluster.

```
spark-submit --class GoogleWebGraphApp /opt/graphx-scripts/target/scala-2.12/graphx-scripts_
```

- `--class GoogleWebGraphApp`: Specifies the main class to run.
- `/opt/graphx-scripts/target/scala-2.12/graphx-scripts_2.12-0.1.jar`: Path to the JAR file you just built.

Step 3: Monitoring and Logs

Once the job is submitted, Spark will process the graph data and calculate the PageRank. You can monitor the job and view logs in the Spark Web UI. The Spark Web UI is accessible at the following ports:

- **Spark Master UI**: `http://localhost:8080`
- **Spark Worker UI (Worker 1)**: `http://localhost:8081`
- **Spark Worker UI (Worker 2)**: `http://localhost:8082`

Sample Logs Here is an example of what the logs might look like when you run the job:

```
SparkContext started!
PageRank results:
(Vertex ID: 1, Rank: 0.015)
(Vertex ID: 2, Rank: 0.023)
(Vertex ID: 3, Rank: 0.014)
(Vertex ID: 4, Rank: 0.017)
...
```

These logs will be printed to the terminal where `spark-submit` is run, showing the computed PageRank scores for each vertex in the graph.

Step 4: Stopping the Spark Session

After the job completes, the Spark session is stopped automatically by the application:

```
spark.stop()
```

This will release resources and stop the Spark context.

Docker Compose File

Here is the relevant Docker Compose configuration for running Spark in a multi-node setup with a master and two workers:

```yaml
services:
  spark-master:
    image: bitnami/spark
    environment:
      - SPARK_MODE=master
    ports:
      - '8080:8080' # Spark Web UI
      - '7077:7077' # Spark Master Port
      - '4000:4040' # Spark History Server
    networks:
      - spark-net
    volumes:
      - ./graphx-scripts:/opt/graphx-scripts

  spark-worker-1:
    image: bitnami/spark
    environment:
      - SPARK_MODE=worker
      - SPARK_MASTER_URL=spark://spark-master:7077
    depends_on:
      - spark-master
    ports:
      - '8081:8081' # Spark Worker UI
    networks:
      - spark-net
    volumes:
      - ./graphx-scripts:/opt/graphx-scripts

  spark-worker-2:
    image: bitnami/spark
    environment:
      - SPARK_MODE=worker
      - SPARK_MASTER_URL=spark://spark-master:7077
    depends_on:
      - spark-master
    ports:
      - '8082:8081' # Second worker with a different port mapping
    networks:
      - spark-net
    volumes:
      - ./graphx-scripts:/opt/graphx-scripts
  sbt-builder:
    image: hseeberger/scala-sbt:11.0.12_1.5.5_2.13.6
    volumes:
      - ./graphx-scripts:/opt/graphx-scripts
    working_dir: /opt/graphx-scripts
    command: sbt package

networks:
  spark-net:
    driver: bridge
```

Conclusion

By following these steps, you can successfully build and submit your Google Web Graph PageRank calculation application to Spark using Docker, SBT, and `spark-submit`. The results will be displayed in the terminal logs and can also be monitored through the Spark Web UI.