Lab3 - Kafka and Spark Streaming

Amir H. Payberah
payberah@kth.se

1 Introduction

In this assignment you will learn how to create a simple Spark Streaming application, while reading streaming data from Kafka. Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams. Spark Streaming provides a high-level abstraction called *discretized stream* or `DStream` that represents a continuous stream of data. `DStream` can be created either from input data streams from sources such as Kafka, Flume, and Kinesis, or by applying high-level operations on other `DStream`. Internally, a `DStream` is represented as a sequence of `RDD`s. In the rest of this document, we first explain how to install and test Kafka, and Cassandra, and then explain the assignment.

2 Installing Kafka

We use Kafka as our data source, which is a distributed, partitioned, replicated commit log service. This section presents the steps you need to do to install Kafka on a Linux machine.

1. Download Kafka from the following link
   https://downloads.apache.org/kafka/3.2.1/kafka_2.12-3.2.1.tgz

2. Set the following environment variables.

```
export KAFKA_HOME="/path/to/the/kafka/folder"
export PATH=$KAFKA_HOME/bin:$PATH
```

3. Kafka uses ZooKeeper to maintain the configuration information, so you need to first start a ZooKeeper server if you do not already have one.

```
$KAFKA_HOME/bin/zookeeper-server-start.sh $KAFKA_HOME/config/zookeeper.properties
```

4. Start the Kafka server.

```
$KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties
```

5. Then, create a *topic*. A topic is a category or feed name to which messages are published. For each topic, the Kafka cluster maintains a partitioned log that looks like this. Let's create a topic named `avg` with a single partition and only one replica.

```
$KAFKA_HOME/bin/kafka-topics.sh --create --topic avg --bootstrap-server localhost:9092 --replication-factor 1
       --partitions 1
```

6. To see the list of topics you can run the following command.

```
$KAFKA_HOME/bin/kafka-topics.sh --list --bootstrap-server localhost:9092
```

7. Test Kafka by produce messages to the `avg` topic and reading those message by a consumer.

```
# Produce messages and send them to the topic "avg"
$KAFKA_HOME/bin/kafka-console-producer.sh --topic avg --bootstrap-server localhost:9092

# Consume the messages sent to the topic "avg"
$KAFKA_HOME/bin/kafka-console-consumer.sh --topic avg --from-beginning --bootstrap-server localhost:9092
```

3 Spark-Kafka Integration

In this assignment, you should test and run the given Spark Streaming application that reads streaming data from Kafka, which are (key, value) pairs in the form of `"String,int"`, and calculates the average value of each key and continuously update it, while new pairs arrive. To do this assignment, first you need to start Kafka and create a topic, named `avg`, as explained in the previous section. In the Spark Streaming code, we use `mapWithState` to calculated the average value of each key in a statful manner, and in the Structure Streaming we use `mapGroupsWithState` to do the same. To test the codes, run `sbt run` in the path of your code.

To generate a streaming input (pairs of `"String,int"`) and feed them to Kafka, you are given a code in the `generator` folder. You just need to run the `sbt run` in its path. Here is an example output that `generator` produces: