Exercise 8:
Apache Spark

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Concepts and Technologies for Distributed Systems and Big Data Processing – SS 2016

Task 1 Paper Reading

Read the paper *Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing* by Zaharia et al. [1], which introduces RDD, the central data structure to Apache Spark, that is maintained in a fault-tolerant way across different machines in a cluster.

You can find the paper at `http://www-bcf.usc.edu/~minlanyu/teach/csci599-fall12/papers/nsdi_spark.pdf`

Task 2 Count Errors

Have a look at the code provided on the course website, which contains the code for CountErrors. The ZIP file contains a SBT and a Maven project, which can be imported into your IDE. An Eclipse project is provided, too. If importing the Eclipse project should not work, please have a look into the informatory section at the end of the exercise to see how to import a Maven project into Eclipse.

Note that, if you are using Eclipse, the IDE must be set up correctly to support Scala projects. Please have a look into the informatory section at the end of the exercise to learn how to setup Eclipse appropriately.

Implement CountErrors, which should count the the number of errors in the given log file. Each line which contains the string `"error"` (case insensitive) counts as one error.

Task 3 Approximate Pi

The project for task 1 provided on the course website also contains the code for ApproximatePi.

Implement ApproximatePi, which should approximate π by creating random samples. Each sample is a randomly chosen point in the unit square (between $(0,0)$ and $(1,1)$). Compute the fraction f of samples which fall into the unit circle (radius of 1 centered at the origin $(0,0)$). Whether a sample falls into the unit circle, can be decided using the Pythagorean theorem.

Since we can assume the samples to be uniformly distributed within the area covered by the unit square and this area is of size 1, f is an approximation for the area covered by the unit circle within the unit square. Since only a quarter of the unit circle falls into the unit square and the complete area A of a circle is given by $A = \pi r^2$, it follows $\pi \approx 4f$, which can be used to compute the approximation of π based on the samples.

Task 4 Reverse Graph

Referring to exercise 3, where ReverseGraph was implemented in Hadoop, complete the following code for ReverseGraph using Spark. ReverseGraph should reverse the direction of the edges in a directed graph.

```
Seq(
  3 -> Seq(1, 2),
  1 -> Seq(2, 3))
```

(a) Input

```
Seq(
  1 -> Seq(3),
  2 -> Seq(1, 3),
  3 -> Seq(1))
```

(b) Expected output.

Figure 1: ReverseGraph

A possible input is given in Figure 1a, where each element in the sequence is a pair which assigns the list of outgoing edges to the nodes in the graph. The expected output is given in Figure 1b. As you can see, for each edge $a \rightarrow b$ in the input there is a corresponding edge $b \rightarrow a$ in the output.

```
1  val sc = new SparkContext
2  val graph = Seq( /* ... */ )
3
4  val reversedGraph =
5
6
7
8
9
10
11
12
13
14
15
```

For this task, you should look at the following Spark operators available on `RDD[T]` objects (`RDD[T]` objects are created by methods such as `sc.textFile(`*path*`)` or `sc.parallelize(`*seq*`)`):

```
def map[U](f: T => U): RDD[U]
def flatMap[U](f: T => TraversableOnce[U]): RDD[U]
```

Setting up Eclipse

In order to run the project in Eclipse, the *Scala IDE for Eclipse* must be installed. If you use the provided Eclipse project, you also need the *Maven Integration for Scala IDE*. Both plug-ins can be installed via *Help* → *Install New Software* and using the following update sites:

Scala IDE for Eclipse: `http://download.scala-ide.org/sdk/lithium/e44/scala211/stable/site`

Maven Integration for Scala IDE: `http://alchim31.free.fr/m2e-scala/update-site/`

Alternatively, you can also create an Eclipse project using *SBT* from the command line by issuing `sbt eclipse` in the root directory of the downloaded project. *SBT* must be installed in that case, of course.

Importing the Maven project into Eclipse

In Eclipse, open the *Import* dialog via *File* → *Import*.
Choose *Existing Maven Projects*.
Choose the root directory of the Maven project containing the `pom.xml` file.
Select the project and click *Finish*.

References

[1] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 2–2, Berkeley, CA, USA, 2012. USENIX Association.