Lecture 22:
- Brief Spark and Dataproc intro

Announcements:
- HW-3 due
- R-3 out
- Project out
- HW-4 out

Apache Spark

Spark is a distributed data processing engine
- can be run standalone (testing) and on clusters (Hadoop, Kubernetes)
- designed for efficient map-reduce like computation ...
- more later
- provides “functional” style library for writing jobs
- employs lazy evaluation, lineage, and checkpointing for fault tolerance

Large number of programmer “built-in” features
- supports many data formats and representations
- provides a data frame and SQL API
- various libraries (e.g., MLlib, streaming, graph analytics)
Apache Spark

Simple data frame (plus SQL functions) examples:

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import *

spark = SparkSession.builder.appName('demo').getOrCreate()
df = spark.createDataFrame([( 'alice', 32), ( 'bob', 28), ...], [ 'name', 'age'])

df.where(col( 'age') <= 30).show()
+--------+
<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>bob</td>
<td>28</td>
</tr>
<tr>
<td>deb</td>
<td>28</td>
</tr>
</tbody>
</table>
...

df.groupBy('age').count().show()
+----+-------+
| age| count |
|----+-------|
| 32 | 2      |
| 28 | 2      |
...
```

© S. Bowers CPSC 324, Spring 2024

Apache Spark

Simple SQL example:

```sql
spark.sql( 'select avg( age ) from {table}', table=df ).show()
+--------+
| avg( age) |
+--------+
| 32.5    |
+--------+

df2 = spark.read.option('header', True).csv( 'people.csv')
df2.show()
+-------+---+
<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice</td>
<td>32</td>
</tr>
</tbody>
</table>
...
```

© S. Bowers CPSC 324, Spring 2024
Apache Spark

Simple RDD example: ... see reading 4

```python
tf = spark.sparkContext.textFile('text.txt')
counts = (tf.flatMap(lambda line: line.split(' ')) 
    .map(lambda word: (word, 1)) 
    .reduceByKey(lambda a, b: a + b) 
)
counts.collect()
```

[( 'When ', 1) , (' beetles ', 1) , ('fight ', 1) , (' and ', 2) , (' poodle ', 2) , ...]

where:

- **lambda** defines an anonymous function (e.g., \( \lambda x \to x + 1 \))
- **flatMap\( f \)** can map an item to many outputs (\( f \) returns a sequence)
- **map\( f \)** is the “normal” version – applies \( f \) to each input
- **reduceByKey\( f \)** aggregates values via \( f \) for each unique key

Google Cloud Dataproc

Dataproc is a service that provides a cluster environment on GCP

- can run a Hadoop or Kubernetes cluster via Dataproc
- can deploy Spark as well as other cluster computing frameworks
- integrated access to other GCP services (Cloud Storage, BigQuery, etc.)

Dataproc is a fully managed hardware and software service

- handles all resource provisioning (based on number and type of machines)
- installs and updates software installed
- supports fairly involved job configurations

⇒ You’ll create some Dataproc clusters for Spark jobs in HW-4
Dataproc Serverless is fully-managed and serverless

- for running Spark jobs without fixing a cluster configuration
- automatically scales the cluster
- charges based on use, not number of vCPUs provisioned (cluster uptime)
- makes setup and maintenance (scaling) much easier

⇒ Recommended for most cases today, also part of HW-4