How to process on distributed large matrix with Spark

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Anis Hamroun

Summary

Goal

Assume A a large matrix such as A included in $R^{1\ 000\ 000\ x\ 1\ 000}$.

We have to compute :

$$A * A^T * A = \sum_{j=1}^{N} dotproduct(Aj, \quad (\sum_{i=1}^{N} outerproduct(Ai, Ai)))$$

The problem is A is too big to be computed with numpy, then we have to choose another way.

Detailed Processus

Step 01:

```
raw_matrix_file = sc.textFile(dataset)
```

We create a RDD from the file.

Step 02:

```
A = raw_matrix_file.map(lambda row: row.split()).map(lambda row: [float(element) for
element in row])
```

Compute A from the first RDD.

Step 03 :

```
AT_A = A.map(lambda row: np.outer(row, row)).reduce(lambda x,y : np.array(x) +
np.array(y))
```

Compute $A^T * A$ from the A.

Assume Ai is the i-ene row of A.

$$A^T * A = \sum_{i=1}^{N} outerproduct(Ai, Ai)$$

https://en.wikipedia.org/wiki/Outer_product

Step 04 :

```
#Compte A * (AT_A)
A_AT_A = A.map(lambda row: np.dot(row,AT_A ))
```

During this step, i compute $A * A^T * A$ from the A.

Assume Ai is the i-ene row of A.

$$A * A^T * A = \sum_{j=1}^{N} dotproduct(Aj, \ (\sum_{i=1}^{N} outerproduct(Ai, Ai)))$$

https://en.wikipedia.org/wiki/Dot_product

Step 05 :

```
A_AT_A.partitionBy(10)
A_AT_A.saveAsTextFile(output_file)
```

A_AT_A is too big to use “collect()” and store it.
Then i split en 10 partitions and i store it.

Performance

With Default configuration.

Data-2-sample.txt	Data-1.txt
11 seconds	5 minutes

My output Matrix is store in this Floader :
“/proj/hamroun/sparkexerice/Spark/matrix/matrix_output2.txt”

github : https://github.com/HappyBearDay/Spark