Informe Speed-Up

Alejandro López Santos

## Introducción:

En este documento vamos a detallar el rendimiento del clúster frente al portátil donde he desarrollado las pruebas, además del rendimiento del servidor en modo clúster o modo single node.

## Preparación de los datos:

Ejecutamos el código varias veces con un *bash script*, que ejecutará el trabajo *Spark* varias veces:

#!/bin/bash  
echo "Realizando SpeedUp Cluster Mode"  
  
for i in {1..100}  
do  
 echo "Ejecucion numero $i"  
 spark-submit --class SparkSQL --master yarn --deploy-mode client --conf spark.dynamicAllocation.enabled=true --conf spark.shuffle.service.enabled=true --conf spark.dynamicAllocation.minExecutors=7 --conf spark.driver.cores=1 --conf spark.driver.memory=6g --conf spark.executor.memory=6g /home/alopez/proyecto/tfg\_scalasparksql\_2.11-0.1.jar "2018-05-04 10:20" "2018-05-04 10:50" cluster\_$i > cluster\_mode\_execution\_$i.txt  
done  
  
echo "Realizando SpeedUp Single Node Mode"  
for i in {1..100}  
do  
 echo "Ejecucion numero $i"  
 spark-submit --class SparkSQL --master yarn --deploy-mode client --conf spark.dynamicAllocation.enabled=true --conf spark.shuffle.service.enabled=true --conf spark.dynamicAllocation.maxExecutors=1 --conf spark.driver.cores=1 --conf spark.driver.memory=6g --conf spark.executor.memory=6g /home/alopez/proyecto/tfg\_scalasparksql\_2.11-0.1.jar "2018-05-04 10:20" "2018-05-04 10:50" single\_$i > single\_node\_$i.txt  
done

Al generar los **.txt** por cada ejecución, y hacer más fácil la obtención de los datos para calcular el tiempo de ejecución de cada iteración, generamos unos archivos más obteniendo el tiempo inicial y obteniendo el tiempo final de cada ejecución:

head -n 1 cluster\_mode\_execution\_\* -q | cut -c1-19 > cluster\_start.txt  
tail -n 1 cluster\_mode\_execution\_\* -q | cut -c1-19 > cluster\_end.txt  
head -n 1 single\_node\_\* -q | cut -c1-19 > single\_start.txt  
tail -n 1 single\_node\_\* -q | cut -c1-19 > single\_end.txt

Y a continuación con el siguiente código en **R** recogemos los cuatro ficheros generados y creamos un *DataFrame* para así hacer la media en segundos de cada ejecución.

# Prepare TXT Output to dataframes  
startDateSingle <- read.csv(  
 file="data/single\_start.csv",  
 header=F  
)  
  
endDateSingle <- read.csv(  
 file="data/single\_end.csv",  
 header=F  
)  
  
# De 100 muestras han fallado la ejecucuón 79, 71, 56, 14 (4 de 100, 96 muestras)  
clusterSingleNodeData <- data.frame(  
 Execution\_Start = startDateSingle$V1,  
 Execution\_End = endDateSingle$V1  
)  
  
startDateCluster <- read.csv(  
 file="data/cluster\_start.csv",  
 header=F  
)  
  
endDateCluster <- read.csv(  
 file="data/cluster\_end.csv",  
 header=F  
)  
  
# De 100 muestras ningún error  
clusterData <- data.frame(  
 Execution\_Start = startDateCluster$V1,  
 Execution\_End = endDateCluster$V1  
)  
  
#Load CSV Data (10 muestras)  
laptopData <- read.csv(  
 file="data/laptopData.csv",  
 header=T,  
 sep=";",  
 stringsAsFactors = T  
)  
  
# Calculate execution times in minutes  
laptopData = as.integer(difftime(  
 as.POSIXlt(laptopData$Execution\_End, format="%Y-%m-%d %H:%M:%S"),   
 as.POSIXlt(laptopData$Execution\_Start, format="%Y-%m-%d %H:%M:%S"),  
 units="secs")  
)  
  
clusterSingleNodeData = as.integer(difftime(  
 as.POSIXlt(clusterSingleNodeData$Execution\_End, format="%Y-%m-%d %H:%M:%S"),   
 as.POSIXlt(clusterSingleNodeData$Execution\_Start, format="%Y-%m-%d %H:%M:%S"),  
 units="secs")  
)  
  
clusterData = as.integer(difftime(  
 as.POSIXlt(clusterData$Execution\_End, format="%Y-%m-%d %H:%M:%S"),   
 as.POSIXlt(clusterData$Execution\_Start, format="%Y-%m-%d %H:%M:%S"),  
 units="secs")  
)

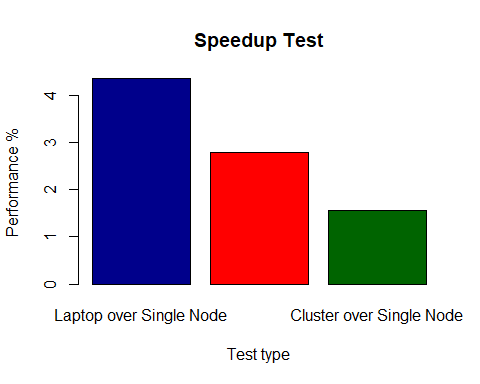
## Fórmula Speedup:

Calculamos el *SpeedUp* con la siguiente fórmula:

# Get the mean execution time, given by 10 execution time samples  
laptopDataMean = mean(laptopData)  
clusterSingleNodeDataMean = mean(clusterSingleNodeData)  
clusterDataMean = mean(clusterData)  
  
dataForPlot <- c(  
 clusterSingleNodeDataMean / laptopDataMean,  
 clusterDataMean / laptopDataMean,  
 clusterSingleNodeDataMean / clusterDataMean  
)

## Visualización de los datos:

barplot (  
 dataForPlot,  
 main = "Speedup Test",  
 xlab = "Test type",  
 ylab = "Performance %",  
 col = c("darkblue", "red", "darkgreen"),  
 names.arg = c("Laptop over Single Node", "Laptop over Cluster", "Cluster over Single Node"),  
 beside = TRUE  
)



## Análisis de los datos:

**Características del clúster –**

**Características del portátil –**

*Procesador*: Intel i5-5200U 2,2ghz 2 núcleos 4 hilos  
*Memoria*: 8gb ram   
*Disco*: 128gb SSD   
*Gráfica*: Nvidia 765m GTX

No es de extrañar que el portátil tenga un mayor rendimiento ya que nuestro trabajo *Spark* hace uso intensivo en disco, leer el *Dataset* y posteriormente escribir el resultado en un CSV. Cada registro tarda en escribirse 10 milisegundos en mi portátil, y en el clúster una media de 200 milisegundos.

Dejando de lado el portátil, se ve la mejoría de 1.78x de pasar de *single node* a modo clúster.