Universidad EAFIT

ST0263: Tópicos Especiales en Telemática, 2017-1

Profesor: Edwin Montoya – [emontoya@eafit.edu.co](mailto:emontoya@eafit.edu.co)

Laboratorio Hadoop

|  |
| --- |
| Instalación de un cluster Hadoop (HDFS+YARN) |
| Ver guía de instalación ([montajedehadoop2.7.2.pdf](https://drive.google.com/open?id=0B_4oKjh0Qca5WllpMTVOclA4cmM))  Monitoreo del cluster hadoop: http://10.131.137.x:8088 |
| Gestión de archivos en HDFS |
| Una vez montado el cluster propio, o utilizando el cluster de producción del DIS (10.131.137.188), con su user y pass de la VPN, realizar:   1. Obtener un subconjunto de datos de prueba, de Gutenberg Digital Library   Puede descargar datos directamente del sitio:  Comando:  $ wget -w 2 -m -H "http://www.gutenberg.org/robot/harvest?filetypes[]=txt&langs[]=es"  asi, los descarga en formato .zip, uds los deben descomprimir antes de enviarlos al HDFS  Puede utilizar datos previamente descargados:  Link (<https://drive.google.com/open?id=0B_4oKjh0Qca5UGxTU3VBNmtxYms> ) |
| Listar archivos HDFS |
| $ hdfs dfs -ls /  $ hdfs dfs -ls /user  $ hdfs dfs -ls /datasets |
| Crear tu propio directorio de usuario en HDFS |
| $ hdfs dfs -mkdir /user/st0263/username  $ hdfs dfs -mkdir /user/st0263/username/data\_in  $ hdfs dfs -mkdir /user/st0263/username/data\_out  reemplace “username” por su usuario asignado. |
| Copiar archivos locales hacia HDFS |
| Se asume que tiene los datos de gutenberg en: ~username/datasets/gutenberg-txt-es o donde los haya descargado.  $ hdfs dfs -copyFromLocal ~username/datasets/gutenberg-txt-es/\*.txt /user/st0263/username/data\_in  otro commando para copier:  $ hdfs dfs -put ~username/datasets/gutenberg-txt-es/\*.txt /user/st0263/username/data\_in  $ hdfs dfs -ls /user/st0263/username/data\_in |
| Copiar archivos de HDFS hacia local |
| $ hdfs dfs -copyToLocal /user/st0263/username/data\_out/out1/\* ~username/data\_out  otro comando para traer:  $ hdfs dfs -get /user/st0263/username/data\_out/out1/\* ~username/data\_out  $ ls -l data\_out/out1 |
| Probar otros commandos |
| Se aplica los siguientes comandos a:  $ hdfs dfs -<command>  comandos:  du <path> uso de disco en bytes  mv <src> <dest> mover archive(s)  cp <src> <dest> copiar archivo(s)  rm <path> borrar archive(s)  put <localSrc> <dest-hdfs> copiar local a hdfs  cat <file-name> mostrar contenido de archivo  chmod [-R] mode cambiar los permisos de un archivo  chown … cambiar el dueño de un archivo  chgrp cambiar el grupo de un archivo |
| PROGRAMACIÓN EN YARN-MAP/REDUCE |
| WordCount en Java  Tomado de: <https://hadoop.apache.org/docs/r2.7.2/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html>  Nativamente, los clusters hadoop corren programas escritos en java.   * Contador de palabras en archivos texto en Java   Se tiene el programa ejemplo: WordCount.java, el cual despues de compilarse en la version hadoop 2.7.3, genera un jar wc.jar, el cual será el que finalmente se ejecute.  El código fuente de WordCount.java, script de generación de jar e instrucciones esta en:  <https://drive.google.com/open?id=0B_4oKjh0Qca5ZHI0WEJGenRZZnM>  Descargarlo, compilarlo y generar el jar (wc.jar)  Para ejecutar:  $ hadoop jar wc.jar WordCount /datasets/gutenberg/es/10814.txt /user/st0263/username/data\_out/out1  el comando hadoop se este abandonando por yarn:  $ yarn jar wc.jar WordCount /datasets/gutenberg/es/10814.txt /user/st0263/username/data\_out/out2 |
| WordCount en Python  Para correr programas python en Yarn-Hadoop, se require una libreria que facilite dicho acceso, para estos ejemplos se empleará mrjob (<https://pythonhosted.org/mrjob/)>  Sino esta instalada la libreria a nivel global o local, debe instalarla:  // global como root o sudo  # yum install python-pip  # pip install mrjob  // python local  $ wget -O virtualenv.py http://bit.ly/virtualenv  $ python virtualenv.py mypython  $ mypython/bin/easy\_install pip  $ mypython/bin/pip install mrjob  wc-mrjob.py  from mrjob.job import MRJob  class MRWordFrequencyCount(MRJob):  def mapper(self, \_, line):  for w in line.decode('utf-8','ignore').split():  yield w,1  def reducer(self, key, values):  yield key, sum(values)  if \_\_name\_\_ == '\_\_main\_\_':  MRWordFrequencyCount.run()  EJECURARLO:  $ python wordcount.py hdfs:///datasets/gutenberg/es/1\*.txt -r hadoop --output-dir hdfs:///user/st0263/username/data\_out/out1  (el directorio /user/st0263/username/data\_out/out1 no debe existir) |
| ADMINISTRACIÓN DEL CLUSTER y YARN |
| Se puede ver mediante web:  <http://10.131.137.x:8088>  o mediante comandos de consola:  $ yarn [comando] [sub-comando] [opciones]  por ejemplo, para listar las aplicaciones ejecutando:  $ yarn application –list  esto lista los app\_id  si quiere eliminar una aplicación:  $ yarn application -kill app\_id |
| APACHE SPARK |
| Spark es una alternativa de mucho mejor rendimiento que Yarn-MapReduce de hadoop  Word count en spark:  wc-spark.py  from pyspark import SparkContext  import sys  sc = SparkContext("local", "Simple WC")  inputdir = sys.argv[1]  outputdir = sys.argv[2]  text\_file = sc.textFile(inputdir)  counts = text\_file.flatMap(lambda line: line.split(" ")) \  .map(lambda word: (word, 1)) \  .reduceByKey(lambda a, b: a + b)  counts.saveAsTextFile(outputdir)  EJECUTARLO:  $ spark-submit wc-spark.py hdfs:///datasets/gutenberg/es/1\*.txt hdfs:///user/st0263/username/data\_out/out1  (el directorio out1 no debe existir, pero data\_out si) |