Работы с Apache Hadoop

Первоначальная настройка рабочего окружения

В начале работы с Apache Hadoop необходим настроить переменные среды окружения (это необходимо выполнить один раз). В своем домашнем каталоге на кластере добавьте в конец файла `~/.bashrc` следующую строку:

```
source /opt/etc/hadoop-vars.sh
```

Отключитесь от кластера и зайдите на головную машину снова.

Работа с файловой системой HDFS

Распределенная файловая система HDFS (Hadoop Distributed File System) предназначена для хранения больших массивов данных. Она же используется для записи полученных в ходе выполнения MapReduce-программ результатов.

Каждый файл в рамках HDFS разбивается на блоки фиксированного размера (128 MiB) и распределяется по вычислительным узлам. Для обеспечения отказоустойчивости каждый блок реплицируется на несколько вычислительных узлов (в текущей конфигурации на 3 узла). Таким образом имеется возможность хранить файлы с размером превышающим размер жесткого диска одного узла.

HDFS можно использовать для хранения рабочих данных большого объема, вместо хранения их в домашней директории (NFS). Для просмотра содержимого HDFS рекомендуется использовать Web-интерфейс.

Для работы с HDFS используется набор команд, похожих по смыслу на файловые команды GNU/Linux.

Список всех команд:

```
$ hdfs dfs -help
```

Выведем на экран содержимое корневого каталога HDFS:

```
$ hdfs dfs -ls /
```

Создадим в домашнем каталоге папку bigdata:

```
$ hdfs dfs -mkdir ./bigdata
```

Скопируем файл с локальной файловой системы в HDFS-каталог bigdata:

```
$ hdfs dfs -put /home/pub/etwiki.xml ./bigdata
```

Скопируем еще один файл с локальной файловой системы в свой HDFS-каталог:

```
$ echo Hello > test.txt
$ hdfs dfs -put ./test.txt ./
```

Выведем содержимое файла test.txt на экран:

```
hdfs dfs -cat ./test.txt
```

Скопируем файл из HDFS-каталога в локальную файловую систему:

```
hdfs dfs -get ./test.txt ./mylocal.txt
```

Удалим файлы из HDFS:

```
hdfs dfs -rm ./test.txt
hdfs dfs -rm -r ./bigdata
```

Компиляция и запуск MapReduce-программ (Java API)

Компиляция MapReduce-программ, созданных с использованием Hadoop Java API:

```
$ javac -classpath `hadoop classpath` WordCount.java
$ jar -cvf wordcount.jar .
```

Далее необходимо загрузить входные данные программы в файловую систему HDFS:

```
$ hdfs dfs -mkdir ./wordcount
$ hdfs dfs -put ~/data.txt ./wordcount/input
```

Запускаем задание (jar-файл):

```
$ hadoop jar ./wordcount.jar ddpcourse.mapred.WordCount \
         -D mapreduce.job.reduces=1 \
         ./wordcount/input ./wordcount/output
```

Выгружаем данные из HDFS в локальную файловую систему

```
$ hdfs dfs -get ./wordcount/output/part* ./result
```

Удаляем результаты (для корректного повторного запуска)

```
$ hdfs dfs -rm -r ./wordcount/output
```

Если данные больше не нужны удаляйте их из HDFS.

Запуск Hadoop Streaming-задач

Процесс работы с задачами Hadoop Streaming подобен описанному выше процессу запуска MapReduce-задач на Java API.

Запуск Hadoop Streamin-задания (для mapper.py и reducer.py):

```
$ hadoop jar \
         /opt/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.3.0.jar \
         -file ./mapper.py -mapper ./mapper.py -file ./reducer.py \
         -combiner ./reducer.py -reducer ./reducer.py \
         -input ./wordcount/input -output ./wordcount/output \
         -numReduceTasks 1
```

Запуск Hadoop Pipes-задач

Собираем нашу C++ программу (wordcount.cpp) с библиотекам Hadoop Pipes:

```
$ g++ -std=c++11 -O2 -I/opt/hadoop/include -c wordcount.cpp \
      -o wordcount.o
$ g++ -o wordcount wordcount.o -L/opt/hadoop/lib/x86_64 \
      -lhadooppipes -lhadooputils -lcrypto -lpthread
```

Исполняемый файл программы (wordcount) необходимо скопировать в файловую систему HDFS:

```
$ hdfs dfs -put ./wordcount ./wordcount/bin
```

Запускаем задание:

```
$ hadoop pipes -D hadoop.pipes.java.recordreader=true \
               -D hadoop.pipes.java.recordwriter=true \
               -program ./wordcount/bin/wordcount -reduces 1 \
               -input ./wordcount/input -output ./wordcount/output
```

Работа с очередями заданий

На кластере настроен планировщик Apache Hadoop CapacityScheduler. По умолчанию всем пользователям разрешено ставить задачи в очередь default.

Список доступных очередей задач:

```
$ hadoop queue -list
```

Список очередей со списком ваших прав доступа к ним:

```
$ hadoop queue -showacls
```

Список заданий (jobs) в очереди default:

```
$ hadoop queue -info default -showJobs
```

Состояние очередей удобнее отслеживать через Web-интерфейс (см. выше).