Cloud Computing HW2

Use Hadoop with Docker Container

R06921105 葉津源

Part 1: Set up Hadoop Docker environment

Steps:

1. create instance
   use mi.xlarge (need more than 8 GB RAM)
2. install docker
   use the following commands to install:

```
sudo apt-get remove docker docker-engine docker.io containerd runc
sudo apt-get update
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo apt-key fingerprint 0EBFCD88
sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
sudo docker run hello-world

```

3. prepare files:

```
git clone https://github.com/sdwangntu/hadoop-cluster.git
cd hadoop-cluster

wget https://archive.apache.org/dist/hadoop/core/hadoop-3.1.2/hadoop-3.1.2.tar.gz
wget https://archive.apache.org/dist/hbase/1.4.9/hbase-1.4.9-bin.tar.gz
## scp to transfer hue-4.3.0.tgz
```

4. set up docker container with hadoop by following commands:

```
docker swarm init
docker network create --driver=overlay --attachable myattachable-network
docker build -t hadoop3cluster .

docker run --hostname=hadoop-master -p 8088:8088 -p 9870:9870 -p 9864:9864 \
    -p 19888:19888 -p 8042:8042 -p 8888:8888 --name hadoop-master \
    --network myattachable-network -d hadoop3cluster

docker run --hostname=hadoop-worker --name hadoop-worker --network myattachable-network -d hadoop3cluster
```

5. now docker container is set up and we can get into the docker containers to run hadoop programs by:

```
docker exec -it hadoop-master bash
docker exec -it hadoop-worker bash
```

Part 2: Run Map Reduce in python for log analysis of apache2 web server access log

Steps: (collaborated with 陳冠甫)

1. write mapper.py

```python
#!/usr/bin/python
import sys
from datetime import datetime
for line in sys.stdin:
        line = line.strip()
        if line == '':
                continue
        begin = line.find('[')
        end = line.find(']')
        record = line[begin+1:end-6]
        record = str(datetime.strptime(record,"%d/%b/%Y:%H:%M:%S"))
        record = record[:len(record)-6]+':00:00'
        print(record+'\t1')

```

   note that on line 6 and 7 we need to handle the case where there is an empty line, otherwise map reduce will fail.

2. write reducer.py

```python
#!/usr/bin/python

from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
for line in sys.stdin:
        line = line.strip()
        word,count = line.split('\t',1)
        try:
                count = int(count)
        except ValueError:
                continue
        if current_word == word:
                current_count += count
        else:
                if current_word:
                        print("%s\t%s" % (current_word,current_count))
                current_count = count
                current_word = word
if current_word == word:
        print("%s\t%s" % (current_word,current_count))

```

3. copy access.log into hadoop
   do
   hdfs dfs -mkdir /input
   hdfs dfs -copyFromLocal access.log /input

4. run Map Reduce
   do
   hdfs dfs -rm -r /output
   hadoop jar /opt/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.1.2.jar -mapper "python $PWD/mapper.py" -reducer "python $PWD/reducer.py" -input "/input" -output "/output"

Results:

2004-03-07 16:00:00	27
2004-03-07 17:00:00	25
2004-03-07 18:00:00	24
2004-03-07 19:00:00	26
2004-03-07 20:00:00	20
2004-03-07 21:00:00	23
2004-03-07 22:00:00	29
2004-03-07 23:00:00	22
2004-03-08 00:00:00	21
2004-03-08 01:00:00	21
2004-03-08 02:00:00	27
2004-03-08 03:00:00	22
2004-03-08 04:00:00	26
2004-03-08 05:00:00	37
2004-03-08 06:00:00	17
2004-03-08 07:00:00	31
2004-03-08 08:00:00	44
2004-03-08 09:00:00	63
2004-03-08 10:00:00	39
2004-03-08 11:00:00	34
2004-03-08 12:00:00	45
2004-03-08 13:00:00	37
2004-03-08 14:00:00	23
2004-03-08 15:00:00	9
2004-03-08 16:00:00	2
2004-03-08 17:00:00	2
2004-03-08 18:00:00	9
2004-03-08 19:00:00	6
2004-03-08 20:00:00	23
2004-03-08 22:00:00	20
2004-03-08 23:00:00	1
2004-03-09 01:00:00	12
2004-03-09 02:00:00	15
2004-03-09 03:00:00	1
2004-03-09 05:00:00	24
2004-03-09 06:00:00	29
2004-03-09 07:00:00	8
2004-03-09 08:00:00	27
2004-03-09 09:00:00	2

2004-03-09 10:00:00 11
2004-03-09 11:00:00 6
2004-03-09 12:00:00 9
2004-03-09 13:00:00 8
2004-03-09 14:00:00 14
2004-03-09 15:00:00 28
2004-03-09 16:00:00 2
2004-03-09 17:00:00 8
2004-03-09 18:00:00 12
2004-03-09 20:00:00 8
2004-03-09 21:00:00 3
2004-03-09 22:00:00 5
2004-03-09 23:00:00 1
2004-03-10 00:00:00 6
2004-03-10 02:00:00 5
2004-03-10 03:00:00 17
2004-03-10 05:00:00 1
2004-03-10 07:00:00 1
2004-03-10 08:00:00 38
2004-03-10 09:00:00 3
2004-03-10 10:00:00 6
2004-03-10 11:00:00 29
2004-03-10 12:00:00 102
2004-03-10 13:00:00 13
2004-03-10 14:00:00 3
2004-03-10 15:00:00 13
2004-03-10 16:00:00 6
2004-03-10 18:00:00 2
2004-03-10 19:00:00 2
2004-03-10 20:00:00 5
2004-03-10 21:00:00 12
2004-03-10 22:00:00 13
2004-03-10 23:00:00 9
2004-03-11 00:00:00 6
2004-03-11 01:00:00 1
2004-03-11 02:00:00 1
2004-03-11 03:00:00 6
2004-03-11 06:00:00 19
2004-03-11 07:00:00 6
2004-03-11 08:00:00 8
2004-03-11 10:00:00 3
2004-03-11 11:00:00 19

2004-03-11 12:00:00	17
2004-03-11 13:00:00	46
2004-03-11 14:00:00	15
2004-03-11 15:00:00	27
2004-03-11 16:00:00	6
2004-03-11 18:00:00	4
2004-03-11 20:00:00	14
2004-03-11 23:00:00	1
2004-03-12 01:00:00	1
2004-03-12 02:00:00	2
2004-03-12 03:00:00	1
2004-03-12 04:00:00	5
2004-03-12 05:00:00	11
2004-03-12 08:00:00	1
2004-03-12 09:00:00	2
2004-03-12 11:00:00	25
2004-03-12 12:00:00	22
2004-03-12 13:00:00	3