Docker-MR.md

# Data Intensive Computing

### CSE 4/587 - Spring 2019

## Docker image for MR over Hadoop and Running a simple Wordcount program

## Overview

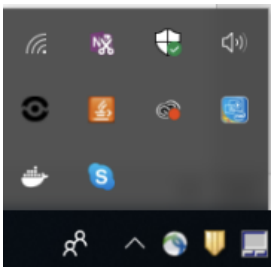
In this document you will learn about downloading and installing a Docker image for Hadoop and execute a simple map-reduce program to illustrate how to run MR programs on Hadoop.

## 1. Install Docker

- Go to docker installation page and select the appropriate OS that you are currently using and follow the instructions

## 2. Increase RAM size for Docker

**On Windows:**
On the right end of your task bar, right click on Docker image → Settings → Advanced and select the RAM size to 8GB. Close and Restart your system. Can you spot the docker icon in the picture below?



**On Mac** On the top of your menu bar you can check docker icon, click on that → Preferences → Advanced → select the RAM size to 8GB. Close and Restart your Docker. Can you spot the docker icon in the picture below?

## 3. Check Installation

- Go to your terminal / powershell and run the following command

```
$ docker run hello-world
```

- You should get the following output, if you don't check your installation:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.
To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
(amd64)
3. The Docker daemon created a new container from that image which runs the
executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
to your terminal.
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

- You can also check the images docker on your machine has by running. You should see `hello-world` image and other images if any in the directory. `$ docker images`

## 4. Pull the Cloudera-Quickstart docker Image

- This docker image manages all your dependencies and you can run all your MapReduce programs easily.
- Run `$ docker pull cloudera/quickstart:latest` this should take a while as the docker container is of about 4 GBs

## 5. Create local directory (on your laptop) for your MR solution and data

- Create a local directory in your C: drive on Windows (for example `C:\Users\bina\Documents` ), ~ or CSE487 or CSE586 on Linux or Mac, and create a directory `dockerMR` for example. This is where you will add your local data, and map and reduce programs of Lab2. This is your local workspace. Later you will map it to your docker and Hadoop file system. The “dockerMR” for example has,

```
mapper.py
reducer.py
data
```

## 6. Configure docker directory and map to local workspace

- Run the following command. If it errors out, restart docker on your machine and then try again.

```
$ docker run --hostname=quickstart.cloudera --privileged=true -t -i -v localpath:/src --publish-
all=true -p 8888 cloudera/quickstart /usr/bin/docker-quickstart
```

replace `localpath` with a location you want to map

**for example**
On a Unix/Linux system:

```
$ docker run --hostname=quickstart.cloudera --privileged=true -t -i -v
/Users/xyz/Documents/SomeFolder:/src --publish-all=true -p 8888 cloudera/quickstart /usr/bin/docker-
quickstart
```

On the Windows Powershell:

```
$ docker run --hostname=quickstart.cloudera --privileged=true -t -i -v
C:\Users\bina\Documents\dockerMR:/src --publish-all=true -p 8888 cloudera/quickstart
/usr/bin/docker-quickstart
```

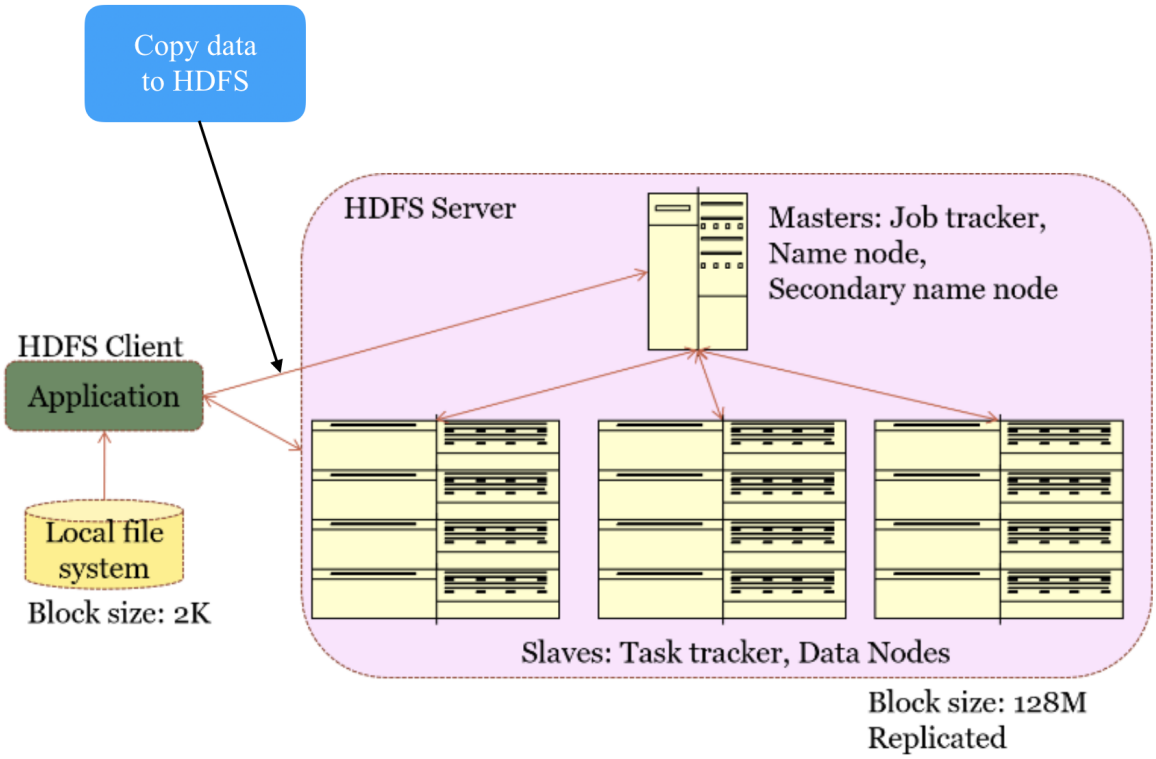
`Note: For windows this will prompt you to share a folder with docker and may ask for your credentials`

- This command will initialize and configure the image and start various frameworks, and result will sequence of outputs. After it completes, it will return a # prompt. To confirm the configuration, do an `ls` , you should see the directory structure shown, where `src` is mapped to your `C:\Users\bina\Documents\dockerMR`
- `$ ls`

```
bin   dev  home  lib64       media  opt           proc  sbin     src  sys  usr
boot  etc  lib   lost+found  mnt    packer-files  root  selinux  srv  tmp  var
```

## 7. Provision Data

- Review the picture we discussed about HDFS.




- To create input locations in your HDFS (name them in a suitable manner) run the following commands

```
$ hadoop fs -mkdir /user/yourName
$ hadoop fs -mkdir /user/yourName/MR
```

```
$ hadoop fs –mkdir /user/yourName/MR/input
```

- These commands create a directory structure for your files. Replace `yourName` with your first name or something like that. Figure above pictures the operations. Copy files from your local file system to Hadoop file system.
- Copy files from your shared folder i.e. `/src/data/` to the HDFS using the following commands, check the directory you created on hadoop fs, and copy data from local to hadoop fs

```
$ cd /src/data/
$ hadoop fs –put file1.txt /user/yourName/MR/input
```

**For multiple files**

```
$ hadoop fs –put *.txt /user/yourName/MR/input
$ hadoop fs –ls /user/yourName/MR/input/
```

Change back to the `/src` directory `$ cd ..`

## 8. Process the data using MR

- Now you are ready to run your MapReduce program using the following command. The hadoop streaming jar the docker provides is located at the following location in the command. If using a Virtual Machine or any other docker or your local machine with hadoop installed you will have to locate the streaming and jar and use that path and name for the jar. Make sure you replace "yourName" in the command below with the same name you used in step 7.

```
$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.6.0-cdh5.7.0.jar \
-file /src/mapper.py    -mapper /src/mapper.py \
-file /src/reducer.py   -reducer /src/reducer.py \
-input /user/yourName/MR/input/* -output /user/yourName/MR/output
```

where `mapper.py` and `reducer.py` are your python programs to perform MapReduce

## 9. Observe the output

- Observe the output generated by MR and move it to your local file system for further processing.

```
$ hadoop fs –cat /user/bina/MR/output/part*
$ hadoop fs –get /user/bina/MR/output/ /src/
```

- Review the output, as we discussed earlier. Once transferred to the local system, you can use it for further processing and extracting information using other algorithms. (You can go back to your dockerMR directory and review the output directory and the MR output.)

## 10. Ready to Quit

- Quit the docker using `Ctrl+P` then `Ctrl+Q`

## 11. Further Explanation

- You can add any number of text files to your input folder and rerun the steps. Also change your MR program ( `mapper.py` and `reducer.py` ) to solve something other than wordcount and apply the steps discussed.

## 12. Additional tips

- If you get a connection refused error when running hadoop commands in the docker, Your name node might not have started or went down. Restart it using `service hadoop-hdfs-namenode restart` command
- `jps` can help you check the health of your hadoop environment
- Read about docker management commands:
    i. `$ docker ps -a`
        - Lists all the running dockers on your system
    ii. `$ docker stop containerID`
        - Stops the docker container running with the given containerID
    iii. `$ docker stop $(docker ps -a -q)`
        - Stops all the docker containers running on your machine
    iv. `$ docker rm $(docker ps -a -q)`
        - Removes the containers from the memory i.e. the docker can't be resumed
    v. `$ docker images`
        - Lists all the images you have on your machine
    vi. `$ docker rmi ImageID`
        - Removes the docker image of the given ImageID from your machine

**Original Document prepared by Yesh Kumar Singh (yeshkuma@buffalo.edu), modified by Bina Ramamurthy (bina@buffalo.edu) to suit the CSE4/587 course**