Pri tej nalogi se bomo spoznali orodjem za distributirano shrambo podatkov Hadoop ter s programskim modelom MapReduce, ki je namenjen obdelavi velikih podatkovnih zbirk paralelno in distributirano na gruči strežnikov.

Najprej moramo vzpostaviti skupino strežnikov, ki bodo imeli nameščeni sistem Apache Hadoop ter bodo tvorili gručo z enim glavnim (master) strežnikom ter tremi podpornimi (slave) strežniki. V ta namen bomo uporabili programsko orodje Docker za pripravo virtualizacije omenjene gruče strežnikov.

Samo programsko orodje najdete na: [Docker](https://download.docker.com/win/stable/Docker%20for%20Windows%20Installer.exe)

Kot osnovno sliko, ki jo bomo uporabili pri virtualizaciji, bomo imeli osnovano na operacijskem sistemu linux:18.04. Samo programsko orodje Docker pa nam bo pri vzpostavitvi virtualizacije posameznega strežnika namestil še željene nadgraditve, ki so v našem primeru Apache Hadoop, YARN, MapReduce ter še podporo HDFS datotečnem sistemu. Vsi podatki glede namestitve so podani v docker datoteki.

[Docker Image](https://github.com/kiwenlau/hadoop-cluster-docker)

###### Based on https://github.com/kiwenlau/hadoop-cluster-docker/blob/master/README.md

##### 1. Build docker

*Za izvedbo ukaza morate biti v direktoriju "..-cluster-docker"*

1. docker build -t gemma/hadoop:1.0 .

##### 2. create hadoop network

1. docker network create --driver=bridge hadoopNetwork

##### 3. Run master

1. docker run -itd --net=hadoopNetwork -p 50070:50070 -p 8088:8088 --name hadoop-master --hostname hadoop-master gemma/hadoop:1.0

##### 4. Run slaves

1. docker run -itd --net=hadoopNetwork --name hadoop-slave1 --hostname hadoop-slave1 gemma/hadoop:1.0
2. docker run -itd --net=hadoopNetwork --name hadoop-slave2 --hostname hadoop-slave2 gemma/hadoop:1.0
3. docker run -itd --net=hadoopNetwork --name hadoop-slave3 --hostname hadoop-slave3 gemma/hadoop:1.0

##### 5. Preveri zagnane instance

1. docker ps

##### 6. Zaženi glavni kontejner

1. docker exec -it hadoop-master bash

##### 6.1 Zaženi/ustavi hadoop

1. sh start-hadoop.sh
2. sh stop-hadoop.sh

##### 7. Stop/Remove all

1. docker stop $(docker ps -a -q)
2. docker rm $(docker ps -a -q)

##### 8. Aplikacija v Javi, ki se požene na Hadoop gruči ter uporabi MapReducer

*Izdelati morate aplikacijo v Javi, ki jo boste naložili na glavni strežnik (master) ter se bo izvedla na načinu MapReduce.*

*Za ta namen naredite Java Maven Projekt v programsko razvojnem okolju (Intellij, Eclipse...) , kjer morate samemu projektu dodati Maven odvnisnosti za podporo MapReduce izvedbi aplikacije.*

*V datoteko s podatki o Maven repozitorijih (pom.xml) dodamo repozitorij za Hadoop Common in Hadoop MapReduce Client Core:*

1. **<dependencies>**
2. **<dependency>**
3. **<groupId>**org.apache.hadoop**</groupId>**
4. **<artifactId>**hadoop-common**</artifactId>**
5. **<version>**2.7.2**</version>**
6. **<scope>**provided**</scope>**
7. **</dependency>**
8. **<dependency>**
9. **<groupId>**org.apache.hadoop**</groupId>**
10. **<artifactId>**hadoop-mapreduce-client-core**</artifactId>**
11. **<version>**2.7.2**</version>**
12. **</dependency>**
13. **</dependencies>**

*Izvedba testnega programa: V direktorij "/src/main/java" dodamo nov razred z imenom "SodaLiha" ter kopiramo naslednjo kodo.*

1. **package** si.gemma;
3. **import** java.io.IOException;
4. **import** java.util.StringTokenizer;
6. **import** org.apache.hadoop.fs.FileSystem;
7. **import** org.apache.hadoop.fs.Path;
8. **import** org.apache.hadoop.io.IntWritable;
9. **import** org.apache.hadoop.io.Text;
10. **import** org.apache.hadoop.mapreduce.Job;
11. **import** org.apache.hadoop.mapreduce.Mapper;
12. **import** org.apache.hadoop.mapreduce.Reducer;
13. **import** org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
14. **import** org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
15. **import** org.apache.hadoop.conf.Configuration;
17. **public** **class** SodaLiha {
18. **public** **static** **class** PreverjanjeSodostiLihosti **extends** Mapper<Object, Text, Text, IntWritable>{
20. **private** **final** **static** IntWritable one = **new** IntWritable(1);
22. **private** Text besedaSoda = **new** Text("Soda");
23. **private** Text besedaLiha = **new** Text("Liha");
25. **public** **void** map(Object key, Text value, Context context) **throws** IOException, InterruptedException {
26. StringTokenizer itr = **new** StringTokenizer(value.toString());
27. **while** (itr.hasMoreTokens()) {
28. **int** stevilka = Integer.parseInt(itr.nextToken());
30. **if**(stevilka % 2 == 0) // soda
31. context.write(besedaSoda, one);
32. **else**
33. context.write(besedaLiha, one);
34. }
35. }
36. }
38. **public** **static** **class** SestejSodeInLihe **extends** Reducer<Text,IntWritable,Text,IntWritable> {
39. **private** IntWritable result = **new** IntWritable();
41. **public** **void** reduce(Text key, Iterable<IntWritable> values, Context context) **throws** IOException, InterruptedException {
42. **int** sum = 0;
43. **for** (IntWritable val : values) {
44. sum += val.get();
45. }
46. result.set(sum);
47. context.write(key, result);
48. }
49. }
51. **public** **static** **void** main(String[] args) **throws** Exception {
53. Configuration conf = **new** Configuration();
54. conf.set("mapred.job.queue.name", "default");
56. // configuration should contain reference to your namenode
57. FileSystem fs = FileSystem.get(**new** Configuration());
58. // true stands for recursively deleting the folder you gave
59. fs.delete(**new** Path(args[1]), **true**);
61. Job job = Job.getInstance(conf, "Stetje Sodih in lihih stevil");
63. job.setJarByClass(SodaLiha.**class**);
64. job.setMapperClass(PreverjanjeSodostiLihosti.**class**);
65. job.setReducerClass(SestejSodeInLihe.**class**);
66. job.setOutputKeyClass(Text.**class**);
67. job.setOutputValueClass(IntWritable.**class**);
69. FileInputFormat.addInputPath(job, **new** Path(args[0]));
70. FileOutputFormat.setOutputPath(job, **new** Path(args[1]));
72. System.exit(job.waitForCompletion(**true**) ? 0 : 1);
73. }
74. }

*Pred build-anjem projekta mu nastavite lastnost, da bo izhodna datoteka artifakt (pri Intellij dodate Artifakte pod izbiro v meniju "File->Project Structure...").*

##### 

##### 9. Dodajanje datoteke na strežnik (containter) ter nato še na HDFS datotečni sistem Hadoopa

*Prijava v glavni strežnik*

1. docker exec -it hadoop-master bash

*Na strežniku kreiramo potrebne poti*

1. mkdir app
2. mkdir testData

*Odjava iz glavnega strežnika*

1. exit

*Dodajanje datoteke na glavni strežni (container)*

1. docker cp LocalTestData/testnestevilke1 hadoop-master:/root/testData
2. docker cp LocalTestData/testnestevilke2 hadoop-master:/root/testData
3. docker cp LocalTestData/SodaLiha.jar hadoop-master:/root/app

*Prijava v glavni strežnik*

1. docker exec -it hadoop-master bash

*Kreiranje novega direktorija na HDFS datotečnem sistemu*

1. hadoop fs -mkdir –p inputData

*Dodajanje datoteke na HDFS datotečni sistem*

1. hdfs dfs -put testData/testnestevilke1 /user/root/inputData
2. hdfs dfs -put testData/testnestevilke2 /user/root/inputData

##### 10. Zagon Java Map Reduce aplikacije

*Poženite testno aplikacijo štetja besed*

1. hadoop jar app/SodaLiha.jar si.gemma.SodaLiha inputData outputData

##### 11. Preverjanje delovanja preko spletnih vmesnikov

*Delovanje strežnika in zagon aplikacije lahko spremljate na lokalnem omrežju na portu* [*8088*](http://localhost:8088/cluster/apps)

*Datotečni sistem vzpostavljene gruče strežnikov lahko vidite na portu* [*50070*](http://localhost:50070/dfshealth.html#tab-overview)

##### 12. Branje izhodnih rezultatov

Poženite testno aplikacijo štetja besed

1. hdfs dfs -cat outputData/part-r-00000

P.S. 1: *Programsko orodje Docker potrebuje za delovanje orodje za virtualizacijo (npr. Oracle Virtual Box, Hyper-V...). V primeru, da uporabljate operacijski sistem Windows ter nimate nameščenega nobenega orodja za virtualizaciji, vam bo Docker sam ponudil možnost uporabe Hyper-V orodja, ki je podprt od Windows 7 (SP1) naprej.*

P.S. 2: *V kolikor pri uporabi Docker-ja prihaja do napake pri povezavi z repozitoriji (Timeout napaka - napako se lahko reproducira z ukazom "docker run hello-world"), poskusite napako rešiti, da prenesete Docker Edge verzijo.* [*Docker*](https://docs.docker.com/docker-for-windows/install/#download-docker-for-windows)