# How to run the experiments

## 1 Settings

In this section will be present how to run the cluster on Docker. The cluster used in the experiments can be found in this repository: docker-spark-cluster.

### 1.1 Number of Nodes

This cluster is composed by 20 nodes. To modify the number of nodes is necessary to edit 2 different files:

- *cluster.sh*: edit the field ”lastNode” with the number of nodes -1;
- */spark/config/workers*: add a number of ”node” + ”*i*”, in base of the number of node that you need.

### 1.2 Hadoop Configuration

In base of the characteristics of the nodes is necessary to modify two different files in the Hadoop configuration. These two file are the following:

- */spark/config/mapred-site.xml*
- */spark/config/yarn-site.xml*

The changes that are necessary to apply regards the following three properties, in both the files:

- *yarn.nodemanager.resource.memory-mb*: that is the amount of physical memory, in MB, that can be allocated for containers. It means the amount of memory YARN can utilize on this node and therefore this property should be lower than the total memory of that machine. The current value is ”2048”.
- *yarn.scheduler.maximum-allocation-mb*: that is the maximum memory allocation available for a container in MB. The current value is ”2048”.
- *yarn.scheduler.minimum-allocation-mb*: that is the minimum unit of RAM to allocate for a Container. The current value is ”128”.

## 2 Build and Deploy the Cluster

To build the docker image is present in *util* folder a file called *execution.sh* that automatically build all the component of the cluster. The execution of this file also allows starting the Namenode and all the defined Datanode.

## 3 Data Upload

To generate the necessary data from TPC-H and the related queries is used the folder *tpch* present in the following repository: Docker-Test-Template.

In this folder are present two *.sh* files:

- *generate_data.sh*: that allow to generate the 8 TPC-H *.tbl* files and move it in the *data* folder.
- *ToDocker.sh*: that allow to copy the data that are generated with the previous file and the 22 TPC-H queries that are present in the *queries* folder, in the following path in the cluster */home/hadoop/dataset/tpch/*.

## 4 Run Test

To run a test in the cluster is necessary to use some files that are present in the folder *test* that is present in the following repository Docker-Test-Template.

### 4.1 Default Test

The *Default Test* is the test that is run without moving the blocks' replicas between the nodes. To execute this type of test is sufficient to execute the file *test.sh.* This file creates the table, load the data into the table, and run the tool to retrieve the files and the nodes information. In this file are also present two *for* cycle. The first one is to execute the 22 queries and the second one indicates the number of times that a single query will be executed. The default value of this second cycle is 5, so each query will be executed 5 times.

NOTE: If the number of nodes that are present in the cluster has been changed, is necessary, before execute the *test.sh* file, to change the *lastNode* parameter with the same value used in section 1.1. Will be necessary to modify the same parameter also in *remove_data.sh* file that is present in *utils* folder.

At the end of each query execution, the file *test.sh* will copy all the important logs information regarding the execution in a folder that it will create in the same folder where the file is.

## 4.2 Algorithm

The optimization algorithm can be found in this repository: Algorithm.

The algorithm takes as input all the information that has been automatically downloaded from the cluster during the execution of a test. This information must be inserted in *data* folder.

The algorithm is composed of two different tools. The first one is the constraint problem-solver, which can be run from the *Algorithm.java* class, which allows executing the algorithm for a fixed amount of time. This amount of time can be changed in *problemsolver/ConstraintsProblemSolverPareto.java* class. After the execution, the algorithm will create a folder called *solutions* that contains some *FilesLocation.txt*. Each of these files is a different possible optimal solution. Theoretically, the last one is the most optimized.

The second tool that composes the algorithm is the benchmark component. This tool can be started from the *benchmark/Benchmark.java* class and use the information contained in the *data/test* folder to generate two important files:

- *benchmark.csv* that contains, for each query, the values of the three metrics evaluated.
- *transferTimes.csv* that contains, for each query, the number of blocks that were moved during query execution.

## 4.3 New Test

The execution of a test that uses one of the solutions that the algorithm has computed can be performed in the same way as the *Default Test*, but is necessary to edit some files before. The first file that must change is *test.sh*, where is necessary to uncomment line number 7 (*#sh utilscopy_data_to_docker.sh*). Then is necessary to move one of the files that the algorithm has computed in *utils* folder and rename it in: *FilesLocation.txt*. Finally can be executed *test.sh* file to perform the queries.