Course NDBI040: **Big Data Management and NoSQL Databases**

Practice 01:

MapReduce

Martin Svoboda

Faculty of Mathematics and Physics, Charles University in Prague

MapReduce: Overview

- **MapReduce**
  - Programming model for processing of large data sets with a parallel and distributed algorithm on a cluster
  - **Map function**
    - Processes input data in order to emit a set of intermediate key/value pairs
    - $(k_1, v_1) \rightarrow \text{list}(k_2, v_2)$
  - **Reduce function**
    - Merges (all) intermediate values associated with the same intermediate key
    - $(k_2, \text{list}(v_2)) \rightarrow (k_2, \text{possibly smaller list}(v_2))$

MapReduce: Example

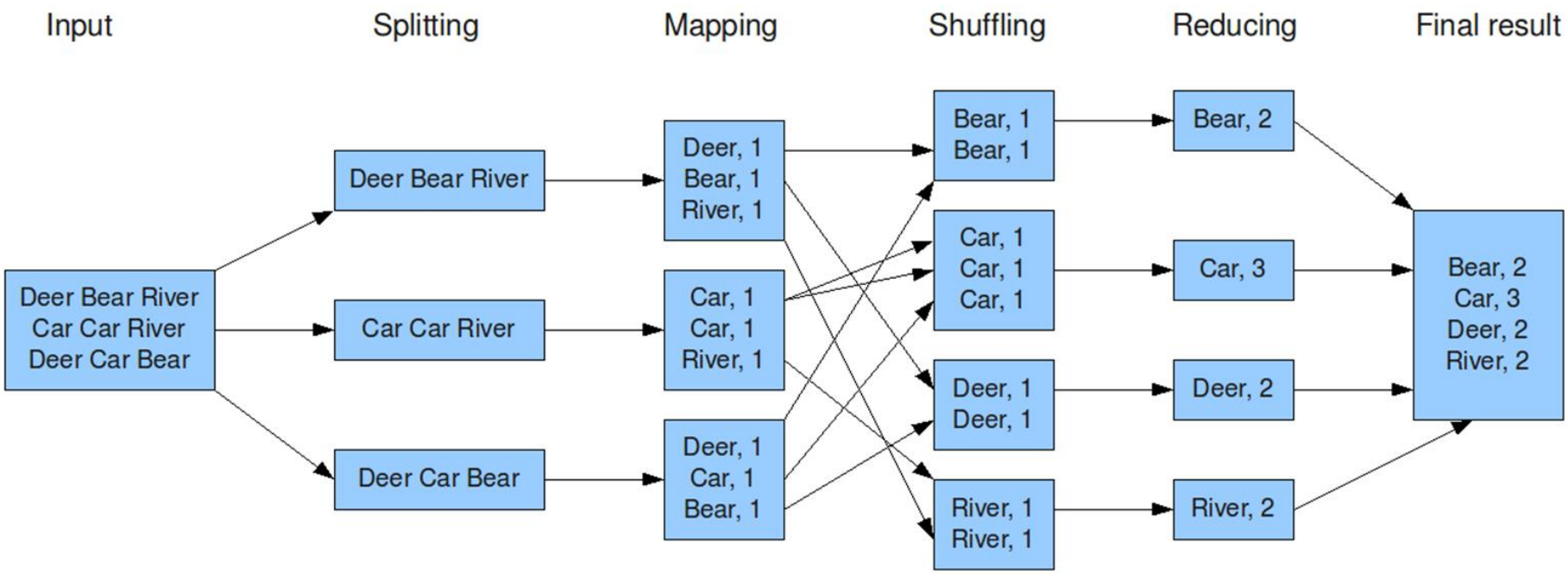
- **Word frequency** – map and reduce functions

```
map(String key, String value):
  // key: document name
  // value: document contents
for each word w in value:
  Emit(w, "1");
```

```
reduce(String key, Iterator values):
  // key: a word
  // values: a list of counts
int result = 0;
for each v in values:
  result += ParseInt(v);
Emit(key, AsString(result));
```

MapReduce: Example

- **Word frequency** – data flow



Hadoop: Overview

- **Apache Hadoop**
  - Open-source software framework allowing to execute applications on large clusters of commodity hardware
  - Modules
    - Hadoop Common
    - **Hadoop Distributed File System (HDFS)**
      - Distributed, scalable, and portable file-system
    - Hadoop YARN
      - Job scheduling and cluster resource management
    - **Hadoop MapReduce**
      - Implementation of the MapReduce programming model

Apache Hadoop Tutorial

HDFS

MapReduce

Hadoop Client

- **SSH and SFTP access**
  - **acheron.ms.mff.cuni.cz:20104**
  - Login: SIS login, lower case (e.g. mylogin)
  - Password: SIS login, upper/lower case (e.g. MyLoGiN)
- Tools
  - **PuTTY**
    - http://www.chiark.greenend.org.uk/~sgtatham/putty/
  - **WinSCP** (Windows Secure Copy)
    - http://winscp.net/

Hadoop Client

- Open PuTTY connection
- Change your password
  - **`passwd`**
- Open WinSCP connection
- Explore directories
  - `/home/mylogin/`
    - Personal directory with private content
  - `/home/NDBI040/`
    - Shared directory with all the tutorial files

First Steps

- Get familiar with basic Hadoop commands
  - **`hadoop`**
    - Help for all Hadoop commands
  - **`hadoop fs`**
    - Hadoop file system commands
  - **`hadoop jar`**
    - MapReduce jobs launching

Word Count Example

- Create your local working directory
  - **mkdir** WordCount
- Make a copy of the sample java source file
  - **cd** WordCount
  - **cp** /home/NDBI040/WordCount.java .
- Compile it
  - mkdir classes
  - **javac** –classpath /usr/lib/hadoop/hadoop-common-2.4.1.jar:/usr/lib/hadoop-mapreduce/hadoop-mapreduce-client-core-2.4.1.jar -d classes/ WordCount.java
  - **jar** -cvf WordCount.jar -C classes/ .

Word Count Example

- Create your HDFS working directory
  - **`hadoop fs -mkdir`** `/tmp/mylogin`
- Prepare sample input data
  - `hadoop fs -mkdir /tmp/mylogin/input1`
  - **`hadoop fs -copyFromLocal`** `/home/NDBI040/file01 /tmp/mylogin/input1/`
  - `hadoop fs -copyFromLocal /home/NDBI040/file02 /tmp/mylogin/input1/`
- Explore the file system (using a web browser)
  - **http://acheron.ms.mff.cuni.cz:20106**
    - (HDFS name node)

Word Count Example

- Run the prepared sample job
  - **`hadoop jar`** `WordCount.jar org.myorg.WordCount /tmp/mylogin/input1 /tmp/mylogin/output1`
- Follow the execution progress (using a web browser)
  - **http://acheron.ms.mff.cuni.cz:20105**
    - (MapReduce JobTracker)

Word Count Example

- Look at the job result
  - **`hadoop fs -copyToLocal`** `/tmp/mylogin/output1/part-00000 ./result.txt`
- Clean the output directory
  - At least in case you want to run the job once again
  - **`hadoop fs -rmr`** `/tmp/mylogin/output1/`

Bigger Example

- Shakespeare
  - **`hadoop fs -mkdir`** `/tmp/mylogin/input2`
  - **`hadoop fs -copyFromLocal`**
    `/home/NDBI040/shake.txt`
    `/tmp/mylogin/input2/shake.txt`
  - **`hadoop jar`** `WordCount.jar org.myorg.WordCount`
    `/tmp/mylogin/input2 /tmp/mylogin/output2`

Useful Commands

- List identifiers of all jobs
  - **`mapred job -list`** `all`
- Kill a particular job
  - **`mapred job -kill`** `<job-id>`
- Print status counters of a job
  - **`mapred job -status`** `<job-id>`

NetBeans Project

- Launch NetBeans IDE
- Create a new project
  - Select *Java application* as project type
- Add Hadoop libraries
  - Make local copies of libraries from `/home/NDBI040/`
    - hadoop-common-2.4.1.jar
    - hadoop-mapreduce-client-core-2.4.1.jar
  - Follow *Add JAR/Folder* and add both these libraries
- Build project to create jar distribution

MapReduce Source Files

- Map and reduce functions
  - `class Map extends MapReduceBase implements Mapper {…}`
  - `class Reduce extends MapReduceBase implements Reducer {…}`
- Key/values pairs
  - `output.collect(key, value);`
  - `WritableComparable` for keys
    `Writable` for values
  - Available classes: `Text`, `IntWritable`, …

MapReduce Source Files

- **Job configuration**
  - Job name
    - `conf.setJobName("WordCount");`
  - Map, reduce and combine functions
    - `conf.setMapperClass(Map.class);`
      `conf.setCombinerClass(Reduce.class);`
      `conf.setReducerClass(Reduce.class);`
  - Types of keys and values
    - `conf.setOutputKeyClass(Text.class);`
      `conf.setOutputValueClass(IntWritable.class);`
  - Input and output format
    - `conf.setInputFormat(TextInputFormat.class);`
      `conf.setOutputFormat(TextOutputFormat.class);`
  - Number of reducers
    - `conf.setNumReduceTasks();`

MapReduce Source Files

- Input and output files
  - `FileInputFormat.setInputPaths(conf, new Path(args[0]));`
    `FileOutputFormat.setOutputPath(conf, new Path(args[1]));`
- Job execution
  - `JobClient.runJob(conf);`
    - Blocks and waits until the job is finished
  - `JobClient.submitJob(conf);`
    - Invokes the job execution but returns immediately
    - Poll for status to make running decisions
    - Or provide a URI to be invoked when the job finishes
      - `conf.setJobEndNotificationURI()`

Local Cluster Network

- RDP (Remote Desktop Protocol) access
  - **vdr.ms.mff.cuni.cz**
  - Login and password as for the client
- Tools
  - **Remote Desktop Connection**
- Cluster
  - Client: **10.20.86.182:22**
  - HDFS name node: **http://10.20.86.180:50070/**
  - MapReduce JobTracker: **http://10.20.86.177:8088/**

References

- **MapReduce tutorial**
  - http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
- **Commands manual**
  - https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/CommandsManual.html
- **File system shell commands**
  - https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html
- **JavaDoc documentation**
  - https://hadoop.apache.org/docs/current/api/

Assignment 1

MapReduce

Assignment 1

- **Select and implement any computation problem that can be solved using MapReduce**
- Store all the related information into your working subdirectory in `/home/NDBI040/`
  - Sample input + output data
  - Java source code + compiled jar
  - Readme file: topic and how-to-run description
- Send an e-mail as a confirmation
  - svoboda@ksi.mff.cuni.cz