Modern Database Systems

Practicals: MapReduce

Doc. RNDr. Irena Holubova, Ph.D.
holubova@ksi.mff.cuni.cz

MapReduce Framework

Recapitulation from the Lecture

- A programming model and associated implementation
- Supports distributed, parallel computing on large data
- Divide-and-conquer paradigm
  - Map processes a key/value pair to generate a set of intermediate key/value pairs
    - $(k_1,v_1) \rightarrow \text{list}(k_2,v_2)$
  - Reduce merges all intermediate values associated with the same intermediate key
    - $(k_2,\text{list}(v_2)) \rightarrow (k_2, \text{possibly smaller list}(v_2))$
- Programmer focuses on map/reduce code
- Framework cares about data partitioning, scheduling execution across machines, handling machine failures, managing inter-machine communication, …

MapReduce

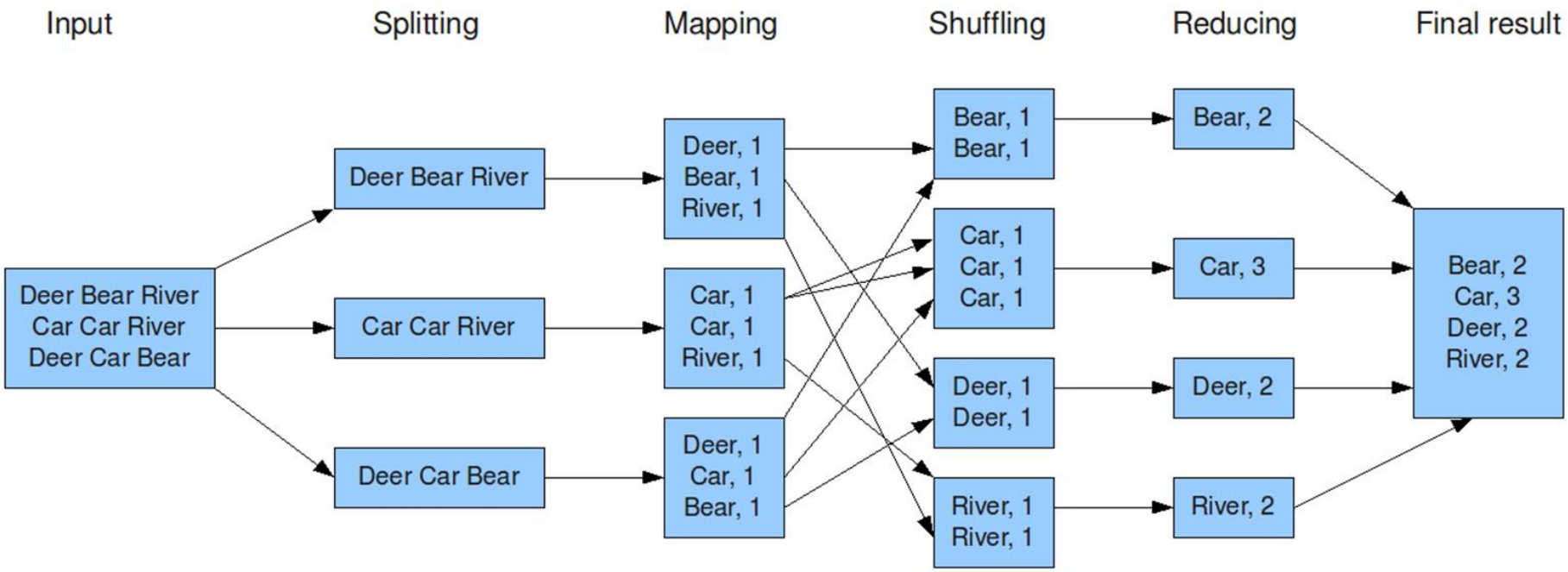
Example: Word Frequency

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

Example – MapReduce

Example: Word Frequency



Apache Hadoop

- Open-source software framework
- Running of applications on large clusters of commodity hardware
  - Multi-terabyte data-sets
  - Thousands of nodes
- Implements MapReduce
- Derived from Google's MapReduce and Google File System (GFS)
  - Not open-source

http://hadoop.apache.org/

Access

Work in the Labs

- shell: SSH nosql.ms.mff.cuni.cz:42222
  - login: mds_<surname>
    - Without diacritic
    - e.g. Jan Novák has: mds_novak
  - password: The same as for video download.
    - Please, change your password!
      - `passwd`
  - Please, do not upload too large data
    - We have a limited disk space and too many users
    - It is not the aim of the practicals
- Sample data shown during practicals can be found in
  - `/home/NDBI040/mapreduce`
  - `/home/NDBI040/mapreduce2`

First Steps

`hadoop`

- Help for Hadoop commands

`hadoop fs`

- Help for HDFS commands

`hadoop jar`

- Execution of MapReduce jobs

```
hadoop fs -ls /
hadoop fs -ls /user/
hadoop fs -ls /user/<login>
```

Note: Copy the commands from
http://www.ksi.mff.cuni.cz/~holubova/NDBI040/slajdy/01_practicals_commands.txt

Testing example in Java

Compile Testing Example `WordCount.java`

- Create your own directory

```
cd ~
mkdir -p mapreduce/WordCount
cd mapreduce/WordCount
```

- Make a copy of the sample java source file

```
cp /home/NDBI040/mapreduce/WordCount.java .
```

Compile Testing Example `WordCount.java`

- Compile sample Word Count implementation

```
mkdir classes
javac -classpath
  /home/NDBI040/mapreduce/hadoop-
  common-
  3.1.1.jar:/home/NDBI040/mapreduce/ha
  doop-mapreduce-client-core-3.1.1.jar
  -d classes/ WordCount.java
jar -cvf WordCount.jar -C classes/ .
```

Store Testing Data to HDFS

- Create your HDFS working directories

```
hadoop fs -mkdir
  /user/<login>/WordCount
hadoop fs -mkdir
  /user/<login>/WordCount/input1
```

- Prepare the sample input data

```
hadoop fs -copyFromLocal
  /home/NDBI040/mapreduce/input1/movie
  s.txt /user/<login>/WordCount/input1
```

Run Testing Example

A small file

```
hadoop jar WordCount.jar WordCount
   /user/<login>/WordCount/input1
   /user/<login>/WordCount/output1
```

- Run the example (it first deletes the output directory `/user/<login>/output`)

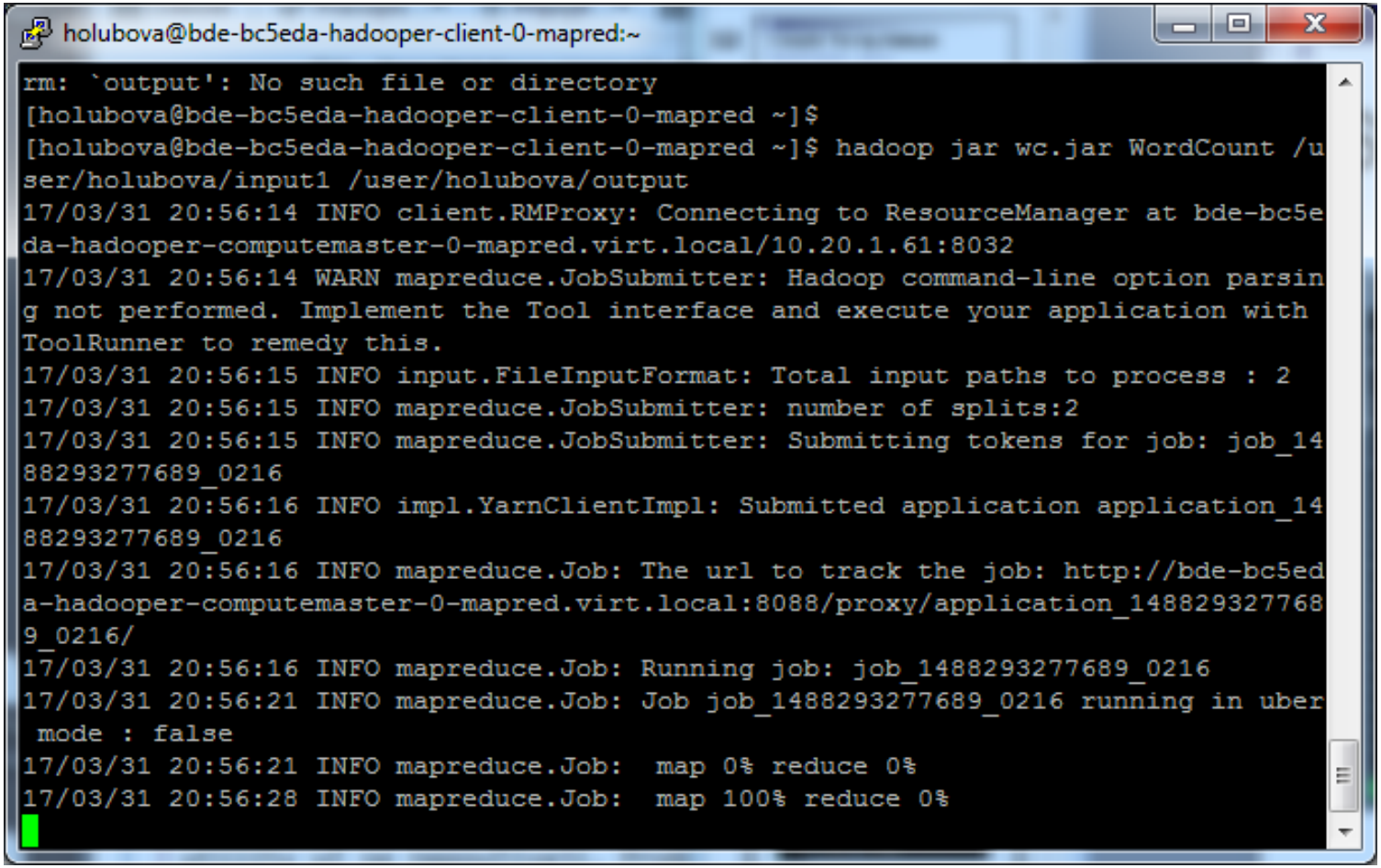
```
hadoop fs -copyToLocal
   /user/<login>/WordCount/output1/part-r-00000
   result.txt
cat result.txt
```

- Check the output

```
hadoop fs -rm -r /user/<login>/WordCount/output1/
```

- Delete the output directory, if required

Status: Command Line



Run Testing Example

Bigger Example

- Create your input2 HDFS directory
- Deploy a copy of the following input:

```
/home/NDBI040/mapreduce/input2/RomeoAndJuliet.txt
```

- Run the MapReduce job
- Retrieve and browse the result
- Clean the output HDFS directory

Useful Commands

```
mapred
```

```
mapred job -list all
```

- List of all jobs

```
mapred job -kill <job-id>
```

- Kill a job

```
mapred job -status <job-id>
```

- Print status (counters)

Testing example in Python

Hadoop MapReduce Program

mapper.py,reducer.py

- Create your own directory

```
cd ~
mkdir -p mapreduce2
cd mapreduce2
```

- Make a copy of the sample source files

```
cp /home/NDBI040/mapreduce2/mapper.py .
cp /home/NDBI040/mapreduce2/reducer.py .
```

Hadoop MapReduce Program

mapper.py,reducer.py

```
hadoop fs -rm -r
  /user/<login>/WordCount/output1/
```

- Delete the output directory, if required

```
hadoop jar
/usr/local/hadoop/share/hadoop/tools/lib/h
adoop-*streaming*.jar
   -files mapper.py,reducer.py
   -mapper mapper.py
   -reducer reducer.py
   -input /user/<login>/WordCount/*
   -output /user/<login>/WordCount/output1
```

- Run the Python example using Hadoop Streaming

Hadoop MapReduce Program

`mapper.py,reducer.py`

```
hadoop fs -ls /user/<login>/WordCount/output1
rm result
hadoop fs -copyToLocal
/user/<login>/WordCount/output1/part-00000
result
cat result
```

- Check and print the result

- For Python experts, there are files `mapper2.py` and `reducer2.py` with a more optimized version

(Optional) Assignment:

- Chose your unique problem domain
  - E.g., the results of football matches of various teams (e.g., one file per week)
    - team A, team B, 24.8. 2022, 7:5
- Think about an original computation problem in your domain which might be solved using MapReduce
  - E.g., return the maximum/average/minimum goals per team, return the sum of points per team (winner = 3 points, draw = each team 1 point), etc.
- Create respective sample data
  - They do not need to be large – this is not the aim of the assignment (the aim is to try to think in the Map/Reduce way)
- Modify the `WordCount.java` sample file respectively

- Points: 2
- Deadline: before the next practicals

References

- Hadoop MapReduce Tutorial: https://hadoop.apache.org/docs/r2.6.0/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
- HDFS commands: https://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-common/FileSystemShell.html