Tugas Besar 2
Big Data IF4044
Menjalankan Map Reduce dengan Spark dari HDFS

Christopher Jeffrey
13520055

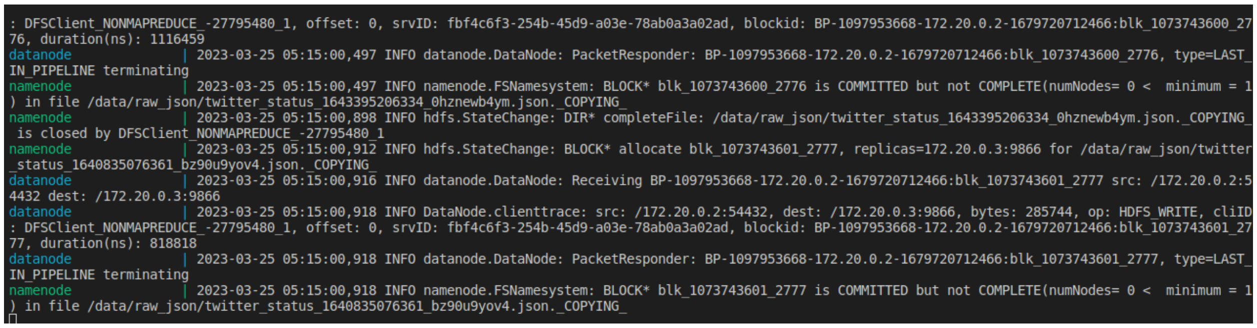
Langkah

Setup

Clone repository https://github.com/christojeffrey/big-data
Jalankan perintah-perintah berikut di terminal

Nyalakan docker

1. `cd tugas-besar-2`
2. `docker compose up`



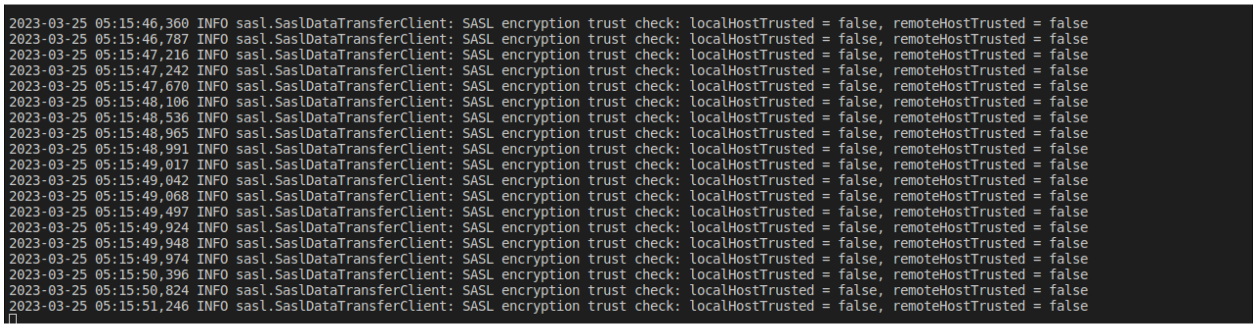
```
: DFSClient_NONMAPREDUCE_-27795480_1, offset: 0, srvID: fbf4c6f3-254b-45d9-a03e-78ab0a3a02ad, blockid: BP-1097953668-172.20.0.2-1679720712466:blk_1073743600_27
76, duration(ns): 1116459
datanode            | 2023-03-25 05:15:00,497 INFO datanode.DataNode: PacketResponder: BP-1097953668-172.20.0.2-1679720712466:blk_1073743600_2776, type=LAST_
IN_PIPELINE terminating
namenode            | 2023-03-25 05:15:00,497 INFO namenode.FSNamesystem: BLOCK* blk_1073743600_2776 is COMMITTED but not COMPLETE(numNodes= 0 <  minimum = 1
) in file /data/raw_json/twitter_status_1643395206334_0hznewb4ym.json._COPYING_
namenode            | 2023-03-25 05:15:00,898 INFO hdfs.StateChange: DIR* completeFile: /data/raw_json/twitter_status_1643395206334_0hznewb4ym.json._COPYING_
 is closed by DFSClient_NONMAPREDUCE_-27795480_1
namenode            | 2023-03-25 05:15:00,912 INFO hdfs.StateChange: BLOCK* allocate blk_1073743601_2777, replicas=172.20.0.3:9866 for /data/raw_json/twitter
_status_1640835076361_bz90u9yov4.json._COPYING_
datanode            | 2023-03-25 05:15:00,916 INFO datanode.DataNode: Receiving BP-1097953668-172.20.0.2-1679720712466:blk_1073743601_2777 src: /172.20.0.2:5
4432 dest: /172.20.0.3:9866
datanode            | 2023-03-25 05:15:00,918 INFO DataNode.clienttrace: src: /172.20.0.2:54432, dest: /172.20.0.3:9866, bytes: 285744, op: HDFS_WRITE, cliID
: DFSClient_NONMAPREDUCE_-27795480_1, offset: 0, srvID: fbf4c6f3-254b-45d9-a03e-78ab0a3a02ad, blockid: BP-1097953668-172.20.0.2-1679720712466:blk_1073743601_27
77, duration(ns): 818818
datanode            | 2023-03-25 05:15:00,918 INFO datanode.DataNode: PacketResponder: BP-1097953668-172.20.0.2-1679720712466:blk_1073743601_2777, type=LAST_
IN_PIPELINE terminating
namenode            | 2023-03-25 05:15:00,918 INFO namenode.FSNamesystem: BLOCK* blk_1073743601_2777 is COMMITTED but not COMPLETE(numNodes= 0 <  minimum = 1
) in file /data/raw_json/twitter_status_1640835076361_bz90u9yov4.json._COPYING_
```

Copy file yang dibutuhkan ke namenode

3. `docker cp raw_json namenode:raw_json`
4. `docker cp filenames.txt namenode:filenames.txt`
5. `docker cp filenames10.txt namenode:filenames10.txt`

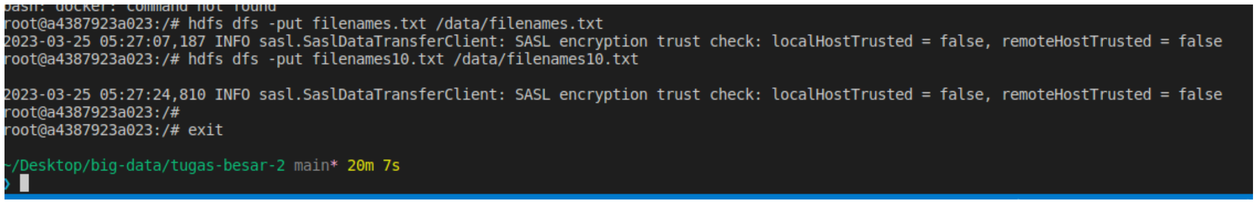
Copy file ke HDFS

6. `docker exec -it namenode bash`
7. `hdfs dfs -mkdir -p /data`
8. `hdfs dfs -put raw_json /data`



```
2023-03-25 05:15:46,360 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:46,787 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:47,216 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:47,242 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:47,670 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:48,106 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:48,536 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:48,965 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:48,991 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,017 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,042 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,068 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,497 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,924 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,948 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:49,974 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:50,396 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:50,824 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2023-03-25 05:15:51,246 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
```

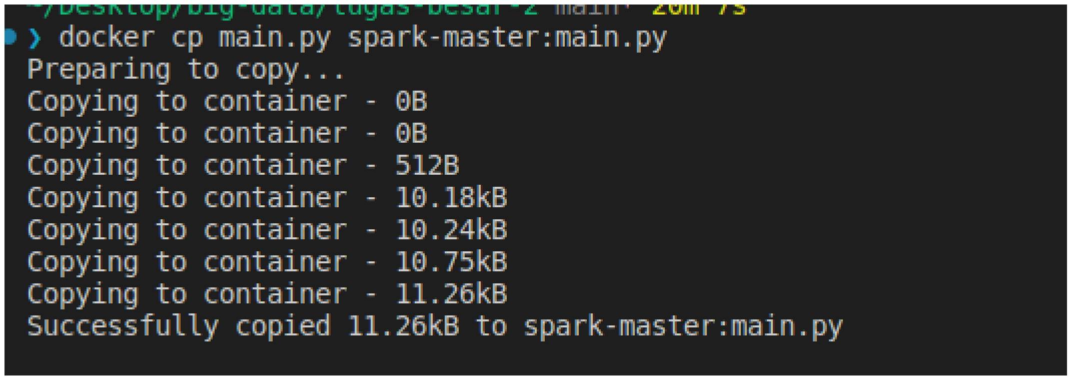
9. `hdfs dfs -put filenames.txt /data/filenames.txt`
10. `hdfs dfs -put filenames10.txt /data/filenames10.txt`
11. Ketik Ctrl D untuk keluar dari docker exec



```
bash: docker: command not found
root@a4387923a023:/# hdfs dfs -put filenames.txt /data/filenames.txt
2023-03-25 05:27:07,187 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
root@a4387923a023:/# hdfs dfs -put filenames10.txt /data/filenames10.txt

2023-03-25 05:27:24,810 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
root@a4387923a023:/#
root@a4387923a023:/# exit

~/Desktop/big-data/tugas-besar-2 main* 20m 7s
❯ █
```

Copy file python untuk map reduce

12. docker cp main.py spark-master:main.py



```
~/Desktop/big-data/tugas-besar-2 main* 20m 7s
❯ docker cp main.py spark-master:main.py
Preparing to copy...
Copying to container - 0B
Copying to container - 0B
Copying to container - 512B
Copying to container - 10.18kB
Copying to container - 10.24kB
Copying to container - 10.75kB
Copying to container - 11.26kB
Successfully copied 11.26kB to spark-master:main.py
```

Jalankan program

13. `docker exec -it spark-master bash`
14. `spark/bin/spark-submit main.py`

Output bisa dilihat di cat output.csv

Mematikan program

15. Untuk mematikan, `docker compose down`

Source Code

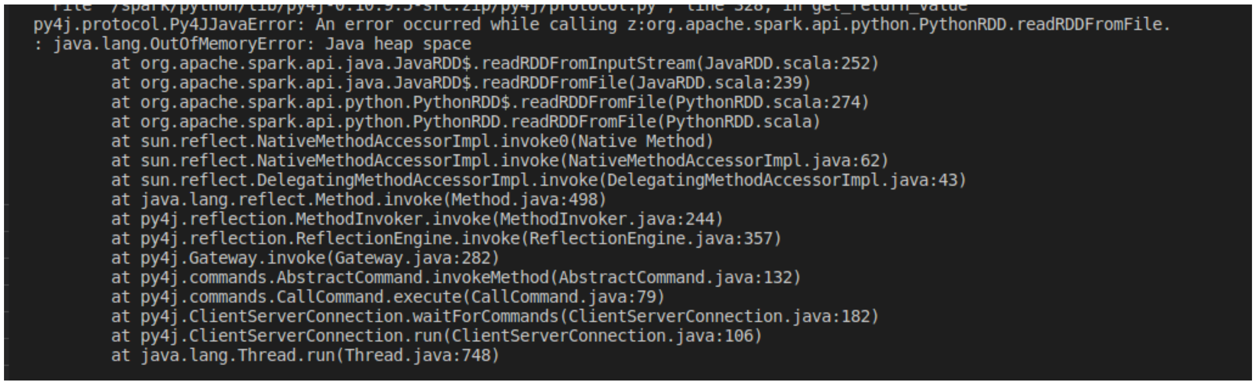
```
FILENAME = "filenames.txt"
print("Reading filenames from " + FILENAME)

from pyspark.sql import SparkSession
import json
import datetime

# functions
def get_file_stating_filename(filename):
    starting_filenames = ['anaktester_go(error)', 'byu.id', 'gridoto_news',
'facebook_post', 'instagram_comment', 'instagram_media', 'instagram_post',
'instagram_status', 'myxl', 'telkomsel', 'twitter_status',
'youtube_comment', 'youtube_video']
    files_starting_filename = ''
    for starting_filename in starting_filenames:
        if starting_filename in filename:
            files_starting_filename = starting_filename
            break
    if files_starting_filename == '':
        raise Exception('files_starting_filename is empty')
    return files_starting_filename

def parse_to_number(df_row):
    filename = df_row[0]
    data = df_row[1]
    try:
        files_starting_filename = get_file_stating_filename(str(filename))
    except Exception as e:
        print("Error parsing filename: " + str(e))
        return None
    try:
        data = json.loads(data)
    except Exception as e:
        print("Error parsing json: " + str(e))
        # print first 100 line
        print("data: " + str(data)[:100])
        return None
    # byu.id
    result = []
    if files_starting_filename == 'byu.id':
        SOCIAL_MEDIA = 'byu.id'
        typenames = data['GraphImages']
        for typename in typenames:
            DATE = typename['taken_at_timestamp']
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            for data in typename["comments"]["data"]:
                DATE = data['created_at']
                # PARSE 1644900422 TO 2021-03-01
                DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
                # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
                result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')

    # gridoto_news
    elif files_starting_filename == 'gridoto_news':
        SOCIAL_MEDIA = 'gridoto_news'
        typenames = data['GraphImages']
        for typename in typenames:
            DATE = typename['taken_at_timestamp']
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            for data in typename["comments"]["data"]:
                DATE = data['created_at']
                # PARSE 1644900422 TO 2021-03-01
                DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
                # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
                result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # facebook_post
    elif files_starting_filename == 'facebook_post':
        SOCIAL_MEDIA = 'facebook_post'
        for datum in data:
            for comments in datum['comments']['data']:
                DATE = comments['created_time']
                # PARSE 2021-03-01T04:00:00+0000 TO 2021-03-01
                DATE = DATE.split('T')[0]
                # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
                result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # instagram_comment
    elif files_starting_filename == 'instagram_comment':
        SOCIAL_MEDIA = 'instagram_comment'
        for datum in data:
            DATE = int(datum['created_time'])
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # instagram_media
    elif files_starting_filename == 'instagram_media':
        SOCIAL_MEDIA = 'instagram_media'
        for datum in data:
            DATE = int(datum['created_time'])
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            COUNT = str(1 + datum["comment"]["count"])
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)
    # instagram_post
    elif files_starting_filename == 'instagram_post':
        SOCIAL_MEDIA = 'instagram_post'
        for datum in data:
            DATE = int(datum['created_time'])
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            COUNT =str(1 + datum["comment"]["count"])
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)

    # instagram_status
    elif files_starting_filename == 'instagram_status':
        SOCIAL_MEDIA = 'instagram_status'
        for datum in data:
            DATE = int(datum['created_time'])
            # PARSE 1644900422 TO 2021-03-01
            DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
            COUNT = str(1 + datum["comment"]["count"])
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + COUNT)
    # myxl
    elif files_starting_filename == 'myxl':
        SOCIAL_MEDIA = 'myxl'
        data = data['GraphImages']
        for datum in data:
            comments = datum['comments']['data']
            for comment in comments:
                DATE = int(comment['created_at'])
                # PARSE 1644900422 TO 2021-03-01
                DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
                # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
                result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # telkomsel
    elif files_starting_filename == 'telkomsel':
        SOCIAL_MEDIA = 'telkomsel'
        data = data['GraphImages']
        for datum in data:
            comments = datum['comments']['data']
            for comment in comments:
                DATE = int(comment['created_at'])
                # PARSE 1644900422 TO 2021-03-01
                DATE =
datetime.datetime.fromtimestamp(DATE).strftime('%Y-%m-%d')
                # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
                result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # twitter_status
    elif files_starting_filename == 'twitter_status':
        SOCIAL_MEDIA = 'twitter_status'
        for datum in data:
            DATE = datum['created_at']
            # Fri Jan 01 05:03:05 +0000 2021 parse to 2021-01-01
            DATE = DATE.split(' ')[5] + '-' + DATE.split(' ')[1] + '-' +
DATE.split(' ')[2]
            DATE = datetime.datetime.strptime(DATE,
'%Y-%b-%d').strftime('%Y-%m-%d')

            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')



        pass
    # youtube_comment
    elif files_starting_filename == 'youtube_comment':
        SOCIAL_MEDIA = 'youtube_comment'
        for datum in data:
            # if doesn't have publishedAt, then skip
            if 'publishedAt' not in datum['snippet']:
                continue
            DATE = datum['snippet']['publishedAt']
            # PARSE 2021-03-01T04:00:00.000Z TO 2021-03-01
            DATE = DATE.split('T')[0]
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
    # youtube_video
    elif files_starting_filename == 'youtube_video':
        SOCIAL_MEDIA = 'youtube_video'
        for datum in data:
            # if doesn't have publishedAt, then skip
            if 'publishedAt' not in datum['snippet']:
                continue
            DATE = datum['snippet']['publishedAt']
            # PARSE 2021-03-01T04:00:00.000Z TO 2021-03-01
            DATE = DATE.split('T')[0]
            # print(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')
            result.append(SOCIAL_MEDIA + '\t' + DATE + '\t' + '1')

    return result


# algorithm
spark = SparkSession.builder.appName("MyApp").getOrCreate()

#  Read in file names
filenames = spark.read.text("hdfs://namenode:9000/data/" + FILENAME)

# 0. setup content list
content = []

for filename in filenames.rdd.collect():
    example = spark.read.text("hdfs://namenode:9000/data/raw_json/" +
filename['value'])

    data = example.rdd.map(lambda x: x['value']).collect(
    )
    data = ''.join(data)
    content.append((filename['value'], data))

# Convert content list to PySpark DataFrame
df = spark.createDataFrame(content, ['filename', 'content'])

# 1. map
mapped = df.rdd.map(parse_to_number).collect()


flatMapped = []
for sublist in mapped:
    # check if sublist is list
    if type(sublist) == list:
        for item in sublist:
            flatMapped.append(item)

print("DONE MAP")

# 2. reduce

counter = {}
for line in flatMapped:
    socialMedia, time, count = line.split("\t")

    if socialMedia not in counter:
        counter[socialMedia] = {}
    if time not in counter[socialMedia]:
        counter[socialMedia][time] = 0

    counter[socialMedia][time] += int(count)

# print
# for socialMedia in counter:
#     for time in counter[socialMedia]:
#         print(socialMedia + "\t" + time + "\t" +
str(counter[socialMedia][time]))


# output to csv
csv = open('output.csv', 'w')
csvString = 'socialMedia,time,count\n'
for socialMedia in counter:
    for time in counter[socialMedia]:
        csvString += socialMedia + "," + time + "," +
str(counter[socialMedia][time]) + "\n"

csv.write(csvString)
```

Program akan membaca sebuah filenames.txt, yang berisi nama-nama file yang berada di dalam folder raw_json. File ini disiapkan sebelumnya. Dari file ini, akan dibuat sebuah list content yang berisi nama file, serta isinya. Content inilah yang akan menjadi input dari mapper. Kegiatan mapping utama dilakukan pada line

```
mapped = df.rdd.map(parse_to_number).collect()
```

Outputnya adalah variable `flatMapped`, berupa list of string, yang akan menjadi input dari reduce.
Reduce dilakukan pada node utama karena terlalu banyak parsing, hanya sekali iterasi saja dari file yang sudah di perpendek, dan tidak perlu membaca hdfs lagi. Hasilnya adalah sebuah dictionary `counter`. Counter dapat dimanipulasi lebih lanjut, misalnya pada kasus ini, disave menjadi sebuah csv.

Untuk input, dibuat dua buah input. Filenames10.txt dan filenames.txt
Jika device tidak cukup mumpuni, mungkin mendapatkan error berikut



```
  File "/spark/python/lib/py4j-0.10.9.5-src.zip/py4j/protocol.py", line 328, in get_return_value
py4j.protocol.Py4JJavaError: An error occurred while calling z:org.apache.spark.api.python.PythonRDD.readRDDFromFile.
: java.lang.OutOfMemoryError: Java heap space
        at org.apache.spark.api.java.JavaRDD$.readRDDFromInputStream(JavaRDD.scala:252)
        at org.apache.spark.api.java.JavaRDD$.readRDDFromFile(JavaRDD.scala:239)
        at org.apache.spark.api.python.PythonRDD$.readRDDFromFile(PythonRDD.scala:274)
        at org.apache.spark.api.python.PythonRDD.readRDDFromFile(PythonRDD.scala)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at py4j.reflection.MethodInvoker.invoke(MethodInvoker.java:244)
        at py4j.reflection.ReflectionEngine.invoke(ReflectionEngine.java:357)
        at py4j.Gateway.invoke(Gateway.java:282)
        at py4j.commands.AbstractCommand.invokeMethod(AbstractCommand.java:132)
        at py4j.commands.CallCommand.execute(CallCommand.java:79)
        at py4j.ClientServerConnection.waitForCommands(ClientServerConnection.java:182)
        at py4j.ClientServerConnection.run(ClientServerConnection.java:106)
        at java.lang.Thread.run(Thread.java:748)
```

Jika memory tidak mencukupi. Oleh karena itu, dapat digunakan filenames10.txt