TDDE31 Lab1

Kristian Sikiric (krisi211) & Pontus Svensson (ponsv690)

1)

```
Highest:                    Lowest:
(u'1975', 36.1)             (u'1990', -35.0)
(u'1992', 35.4)             (u'1952', -35.5)
(u'1994', 34.7)             (u'1974', -35.6)
…                           (u'1954', -36.0)
(u'1960', 29.4)             …
(u'1950', 29.4)             (u'1967', -45.4)
(u'1998', 29.2)             (u'1987', -47.3)
(u'1951', 28.5)             (u'1978', -47.7)
(u'1965', 28.5)             (u'1999', -49.0)
(u'1962', 27.4)             (u'1966', -49.4)
```

a)

```
Highest (with station number):     Lowest (with staion number):
(u'1975', (36.1, u'86200'))        (u'1990', (-35.0, u'166870'))
(u'1992', (35.4, u'63600'))        (u'1952', (-35.5, u'192830'))
(u'1994', (34.7, u'117160'))       (u'1974', (-35.6, u'166870'))
…                                  (u'1954', (-36.0, u'113410'))
(u'1950', (29.4, u'75040'))        …
(u'1960', (29.4, u'173810'))       (u'1967', (-45.4, u'166870'))
(u'1998', (29.2, u'63600'))        (u'1987', (-47.3, u'123480'))
(u'1951', (28.5, u'75040'))        (u'1978', (-47.7, u'155940'))
(u'1965', (28.5, u'116500'))       (u'1999', (-49.0, u'192830'))
(u'1962', (27.4, u'76380'))        (u'1966', (-49.4, u'179950'))
```

b) Non-parallelizied: For small file; 267 seconds, for big file; 2120 seconds.

The spark version was considerably faster, 86 seconds for the small file and a couple of minutes for the big file. The spark version uses parallel computation which probably is the reason it is faster than the non-parallelized program.

2)

Number of readings for each month:

```
(u'2008', u'11', 1663)
(u'1970', u'01', 1)
(u'1971', u'06', 47448)
(u'1989', u'12', 32)
(u'1966', u'11', 213)
…
```

Number of distinct readings:

```
(u'2004', u'06', 153)
(u'2009', u'04', 16)
(u'1973', u'10', 24)
(u'1984', u'04', 14)
(u'1991', u'09', 69)
...
```

3)

```
(u'1971', u'10', u'71470', 8.901612903225805)
(u'1992', u'07', u'178850', 9.80483870967742)
(u'1990', u'03', u'76160', 6.104838709677419)
(u'1963', u'12', u'97200', -2.693548387096774)
(u'2010', u'04', u'93520', 3.961290322580644)
…
```

4)

The result for this assignment was empty.

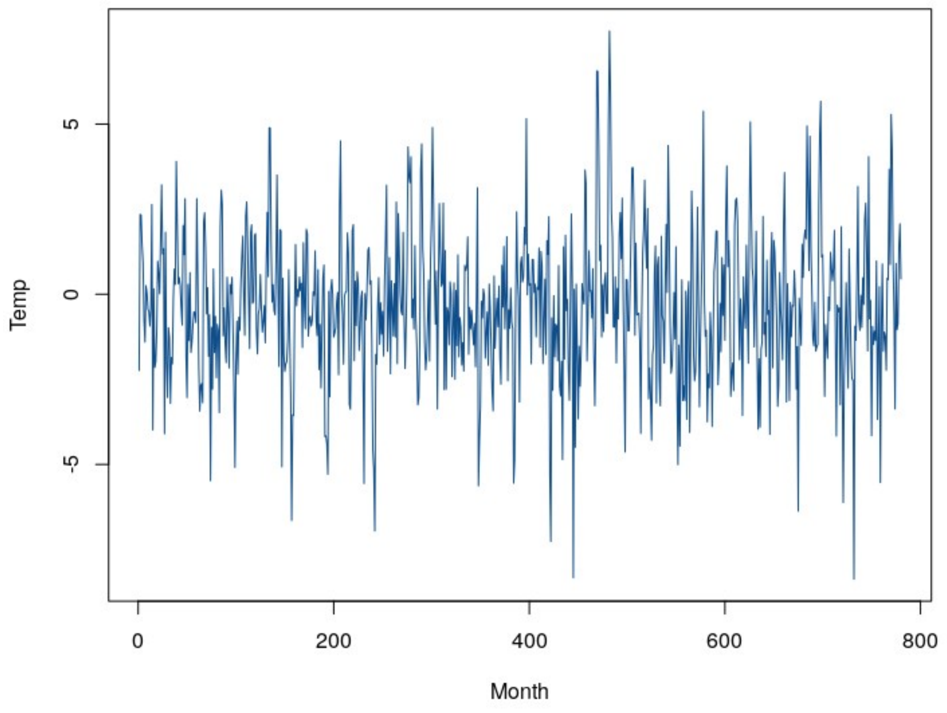
5)

```
(u'2008', u'05', 4.082352941176464)

(u'1998', u'07', 15.029411764705932)

(u'1996', u'12', 6.979411764705853)

(u'1993', u'04', 0.0)

(u'1996', u'07', 14.829411764705902)

…
```

6)

```
(u'02', u'1974', 4.413315683772325)

(u'02', u'1955', -2.0995875420341283)

(u'02', u'2002', 5.0617027805465185)

(u'02', u'1991', 1.0407350386110341)

(u'02', u'1986', -4.857652058163159)

…
```

The following plot shows the difference in temperature between the average temperautre for a month and the long-term average for that month.



Appendix

1a

```
from pyspark import SparkContext

def max_temp(a,b):
        if a[0] >= b[0]:
                return a
        else:
                return b


def min_temp(a,b):
        if a[0] <= b[0]:
                return a
        else:
                return b

sc = SparkContext(appName = "lab1-1")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temp = lines.map(lambda x: (x[1][0:4], (float(x[3]),x[0])))
year_temp = year_temp.filter(lambda x: int(x[0]) >= 1950 and int(x[0]) <= 2014)
max_temps = year_temp.reduceByKey(max_temp)
min_temps = year_temp.reduceByKey(min_temp)
max_temps_sorted = max_temps.sortBy(ascending = False, keyfunc=lambda k: k[1][0])
min_temps_sorted = min_temps.sortBy(ascending = False, keyfunc=lambda k: k[1][0])
max_temps_sorted.saveAsTextFile("max_temps")
min_temps_sorted.saveAsTextFile("min_temps")
```

1b

```
import csv
import json
import time

start = time.time();

lines = list()
line = tuple()
year_temp = list()
max_temps = dict()
print("Start of file")
with open("/nfshome/hadoop_examples/shared_data/temperatures-big.csv", "rb") as csvfile:
        print("Opening")
        for row in csvfile:
                row = row.split(";")
                line = (row[1][0:4], (float(row[3])))
                if(int(line[0]) >= 1950 and int(line[0]) <= 2014):
                        if(line[0] in max_temps):
                                if (line[1] > max_temps[line[0]]):
                                        max_temps[line[0]] = line[1]
                        else:
                                max_temps[line[0]] = line[1]
print("Opened")
max_temps_sorted = sorted(max_temps.items(), key = lambda kv: kv[1], reverse = True)
with open("max_temps_b.txt","w") as file:
        file.write(json.dumps(max_temps_sorted))
file.close()

print(time.time()-start)
#267 seconds for small file
#2120 seconds for big
```

2a

```
from pyspark import SparkContext

def count(a,b):
        return len(a)

sc = SparkContext(appName = "lab1-2")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temp = lines.map(lambda x: ((x[1][0:4], x[1][5:7]), float(x[3])))
year_temp = year_temp.filter(lambda x: int(x[0][0]) >= 1950 and
                                int(x[0][0]) <= 2014 and x[1] >= 10)

temp_count = year_temp.groupByKey()
temp_count=temp_count.map(lambda x: (x[0][0], x[0][1],len(x[1])))

temp_count.saveAsTextFile("count_temps")
```

2b

```
from pyspark import SparkContext

sc = SparkContext(appName = "lab1-2")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temp = lines.map(lambda x: ((x[1][0:4], x[1][5:7],x[0]), float(x[3])))
year_temp = year_temp.filter(lambda x: int(x[0][0]) >= 1950 and
                                    int(x[0][0]) <= 2014 and x[1] >= 10)

temp_count = year_temp.groupByKey()
temp_count=temp_count.map(lambda x: (x[0][0], x[0][1],len(x[1])))

temp_count.saveAsTextFile("count_temps_2")
```

3

```
from pyspark import SparkContext

def average(a):
        return max(a) + min(a)

sc = SparkContext(appName = "lab1-2")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temp = lines.map(lambda x: ((x[1][0:4], x[1][5:7], x[1][8:10],x[0]), float(x[3])))
year_temp = year_temp.filter(lambda x: int(x[0][0]) >= 1960 and
                                                     int(x[0][0]) <= 2014)

temp_count = year_temp.groupByKey()
temp_count=temp_count.map(lambda x: ((x[0][0], x[0][1],x[0][3]),average(x[1])))
temp_count = temp_count.groupByKey()
temp_count=temp_count.map(lambda x: (x[0][0], x[0][1],x[0][2],sum(x[1])/62))

temp_count.saveAsTextFile("count_temps_3")
```

4

```
from pyspark import SparkContext

sc = SparkContext(appName = "lab1-4")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
precipitation_file = sc.textFile("/user/x_krisi/data/precipitation-readings.csv")
tempLines = temperature_file.map(lambda line: line.split(";"))
rainLines = precipitation_file.map(lambda line: line.split(";"))
station_temp = tempLines.map(lambda x: (x[0], float(x[3])))
station_temp = station_temp.groupByKey()
station_temp = station_temp.map(lambda x: (x[0],max(x[1])))
station_temp = station_temp.filter(lambda x: x[1] >= 25 and x[1] <= 30)

station_rain = rainLines.map(lambda x: ((x[0],x[1][0:4], x[1][5:7], x[1][8:10]), float(x[3])))
station_rain = station_rain.groupByKey()
station_rain = station_rain.map(lambda x: (x[0][0],sum(x[1])))
station_rain = station_rain.filter(lambda x: x[1] >= 100 and x[1] <= 200)

joined = station_temp.join(station_rain)



joined.saveAsTextFile("lab4")
```

5

```
from pyspark import SparkContext

sc = SparkContext(appName = "lab1-5")
precipitation_file = sc.textFile("/user/x_krisi/data/precipitation-readings.csv")
station_file = sc.textFile("/user/x_krisi/data/stations-Ostergotland.csv")

rainLines = precipitation_file.map(lambda line: line.split(";"))
stationLines = station_file.map(lambda line: line.split(";"))

station = stationLines.map(lambda x: int(x[0]))
station = station.collect()
broadcastVar = sc.broadcast(station)
rain = rainLines.filter(lambda x: int(x[0]) in broadcastVar.value)
rain = rain.map(lambda x: ((x[1][0:4], x[1][5:7]), float(x[3])))
rain = rain.filter(lambda x: int(x[0][0]) >= 1993 and
                                                            int(x[0][0]) <= 2016)
rain = rain.groupByKey()
tot_rain = rain.map(lambda x: (x[0][0],x[0][1],sum(x[1])/len(station)))

tot_rain.saveAsTextFile("lab1-5")
```

6

```
from pyspark import SparkContext

def average(a):
        return max(a) + min(a)

sc = SparkContext(appName = "lab1-6")
temperature_file = sc.textFile("/user/x_krisi/data/temperature-readings.csv")
station_file = sc.textFile("/user/x_krisi/data/stations-Ostergotland.csv")

tempLines = temperature_file.map(lambda line: line.split(";"))
stationLines = station_file.map(lambda line: line.split(";"))

station = stationLines.map(lambda x: int(x[0]))
station = station.collect()
broadcastVar = sc.broadcast(station)
temp = tempLines.filter(lambda x: int(x[0]) in broadcastVar.value)
temp = temp.map(lambda x: ((x[1][0:4],x[1][5:7],x[1][8:10]), float(x[3])))

#Long term avg
long_temp = temp.filter(lambda x: int(x[0][0]) >= 1950 and
                                                                      int(x[0][0]) <= 1980)
long_temp = long_temp.map(lambda x: (x[0][1], x[1]))
long_temp = long_temp.groupByKey()
long_temp = long_temp.map(lambda x: (x[0],sum(x[1])/len(x[1])))
print(len(temp.collect())) #2648795

#avg temp
temp = temp.filter(lambda x: int(x[0][0]) >= 1950 and
                                                                      int(x[0][0]) <= 2014)
temp = temp.groupByKey()
temp=temp.map(lambda x: ((x[0][0], x[0][1]),average(x[1])))
temp = temp.groupByKey()
avg_temp=temp.map(lambda x: (x[0][1],(x[0][0],sum(x[1])/62)))

#Difference
test = avg_temp.join(long_temp)
test = test.map(lambda x: (x[0],x[1][0][0],(x[1][0][1]-x[1][1])))

test.saveAsTextFile("lab1-6")
```