Zad2

November 3, 2020

[1]:
```
import itertools as it
from pyspark import SparkConf, SparkContext
from operator import add
```

[2]:
```
sc = SparkContext('local[*]')
file = sc.textFile('2.txt')
```

[3]:
```
def split_line(line):
    user, friends = line.split('\t', 1)
    return (user, [] + friends.split(','))

user_friends = file.map(lambda line: split_line(line))
# user_friends.take(5)
```

[4]:
```
user_list = sorted(user_friends.map(lambda row: int(row[0])).collect())
# user_list
```

[5]:
```
def map_frienships(user_friends):
    user, friend_list = user_friends[0], user_friends[1]

    user_friend_pairs = [((user, fr), float('-inf')) for fr in friend_list]
    possible_frend_pairs = [((fr1, fr2), 1) for fr1, fr2 in it.
 ↪permutations(friend_list, 2)]

    return user_friend_pairs + possible_frend_pairs

frienship_pairs = user_friends.flatMap(map_frienships)
# frienship_pairs.take(5)
```

[6]:
```
recommendation_count = frienship_pairs.reduceByKey(add) \
                                      .filter(lambda val: val[1] > 0)
# recommendation_count.take(5)
```

[7]:
```
def map_to_user_recommendation(pair):
    users, mutual_count = pair[0], pair[1]
    u1, u2 = users[0], users[1]
    return (int(u1), (int(u2), mutual_count))
```

```
def top_recommendations(user_recs):
    return sorted(user_recs, key=lambda rec: (-rec[1], rec[0]))[:10]

user_recommendations = recommendation_count.map(map_to_user_recommendation) \
                                           .groupByKey() \
                                           .mapValues(top_recommendations)
#                                            .sortByKey()
# user_recommendations.take(5)
```

[8]:
```
def parse_recommendation(user_recs):
    user, recs = user_recs[0], user_recs[1]
    rec_list = [str(rec[0]) for rec in recs]
    return (user, ', '.join(rec_list))

parsed_recommendations = user_recommendations.map(parse_recommendation).
 ↪collect()
# parsed_recommendations
```

[9]:
```
rec_map = {rec[0]:rec[1] for rec in parsed_recommendations}

def get_recommendation(user):
    recs = rec_map.get(user, '')
    return str(user) + '\t' + recs

result = [get_recommendation(user) for user in user_list]
```

[10]:
```
with open("result.txt", "w") as outfile:
    outfile.write("\n".join(result))
```

[11]:
```
sc.stop()
```