EE412 Foundation of Big Data Analytics, Fall 2019
HW1

Name: 오승준

Student ID: 20160381

Discussion Group (People with whom you discussed ideas used in your answers):
손채연(Problem1 - algorithm, Problem3 – LSH concept)

On-line or hardcopy documents used as part of your answers:

Answer to Problem 1

(a) Find potential friends in a social network using Spark.

< Algorithm >

- Design

  Pair 를 Key 로 갖고, 1 혹은 0 을 value 로 갖는 구조를 이용하였다.

  Line 이 '0 1,2,3,4,5'로 이루어져 있다면, 0 의 친구들 사이의 pair 는 0 을 공통 친구로 갖는다. 따라서 이 점을 이용하여 line 을 입력 받을 때 가능한 pair 를 모두 만들었다. 단, pair 의 user2 가 user1 보다 큰 경우에 대해서만 만들었다.

  그리고, value 에는 친구인지 혹은 우리가 찾고자 하는 interest 가 될 수 있는지에 따라서 0 과 1 로 나누어 넣어주었다. Pair 가 친구이면 value 로 0 을 갖고, 친구가 아닐 수 있으면 1 을 넣었다.

- Map & Reduce

  Map 은 위의 Design 에서 언급한대로 (pair, value)로 mapping 되도록 하였다. 그리고 Reduce 부분에서 친구를 걸러내는 과정을 추가하였다. 만약 친구라면, value 를 더하는게 아니라 계속 0 으로 유지하여 나중에 filter 가 가능하도록 하였다.

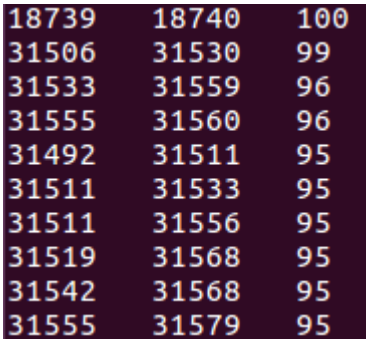
- Sort

  Sort 는 takeOrdered 를 이용하였다. reduceByKey 이후, value 는 각 pair 가 갖는 공통 친구의 수로 되어있다. 이를 이용하여, value 가 큰 순서대로 10 개를 얻도록 했다.

Value 순서대로 10 개를 얻은 이후에는, value 가 같은 경우를 고려하여 key 에 대해서도 sorting 을 하였다.

< Output >

- Run-time: 약 3 분
- Output



18739	18740	100
31506	31530	99
31533	31559	96
31555	31560	96
31492	31511	95
31511	31533	95
31511	31556	95
31519	31568	95
31542	31568	95
31555	31579	95

Answer to Problem 2

(a) Solve the following problems which are based on the exercises in the MMDS textbook.

1. triangular-matrix 를 사용하는 경우

   N 개의 frequent item 으로 pair 를 만들고, pair 의 개수를 세어야 한다. 이 때 만들어지는 triangular-matrix 는 N*N 의 upper-triangle matrix 이다. 따라서 총 $\frac{N\times(N-1)}{2}$ 만큼의 pair 가 만들어지고, pair 당 integer 하나를 저장하므로 S 에 대한 식은 다음과 같다.

$$S = \frac{N \times (N-1)}{2} \times 4 = 2 \times N \times (N-1)$$

2. item-item-count triple 을 사용하는 경우

   Triple 을 사용하면, frequent pair 수만큼 triple 을 저장하여야 한다. Triple 은 100 만개의 frequent pair 와 2M 개 중 frequent item 으로만 이루어진 M 개를 만들어야 한다. Triple 은 interger 3 개를 저장해야 하므로 S 에 대한 식은 다음과 같다.

$$S = (10^6 + M) * 3 * 4 = 12 \times (10^6 + M)$$

   최종적인 S 는 1 번에서 구한 S 와 2 번에서 구한 S 중 최솟값이다.

(b) Find frequent itemsets using the A-Priori algorithm.

< Output >

- Run-time: 약 1 분
- Output

```
20160381@eelab13:~/ee412-bigdata/hw1$ date
Wed Oct  2 22:23:30 KST 2019
20160381@eelab13:~/ee412-bigdata/hw1$ python hw1_2.py browsing.txt
363
328
ELE17451        DAI62779        1592
FRO40251        SNA80324        1412
FRO40251        DAI75645        1254
FRO40251        GRO85051        1213
GRO73461        DAI62779        1139
SNA80324        DAI75645        1130
DAI62779        FRO40251        1070
DAI62779        SNA80324        923
DAI62779        DAI85309        918
GRO59710        ELE32164        911
20160381@eelab13:~/ee412-bigdata/hw1$
20160381@eelab13:~/ee412-bigdata/hw1$
20160381@eelab13:~/ee412-bigdata/hw1$ date
Wed Oct  2 22:24:29 KST 2019
```

Answer to Problem 3

(a) Solve the following exercises in the MMDS textbook

a. 2-way AND, 3-way OR

초기: p

2-way AND 이후: p^2

3-way OR 이후: $1 - (1 - p^2)^3$

b. 3-way OR, 2-way AND

초기: p

3-way OR 이후: $1 - (1 - p)^3$

2-way AND 이후: $(1 - (1 - p)^3)^2$

c. 2-way AND, 2-way OR, 2-way AND

2-way AND 이후: p^2

2-way OR 이후: $1 - (1 - p^2)^2$

2-way AND 이후: $(1 - (1 - p^2)^2)^2$

d. 2-way OR, 2-way AND, 2-way OR

2-way OR 이후: $1 - (1 - p)^2$

2-way AND 이후: $(1 - (1 - p)^2)^2$

2-way OR 이후: $1 - (1 - (1 - (1 - p)^2)^2)^2)^2$

(b) Find similar documents using minhash-based LSH

< Output >

- Run-time: 약 2 분
- Output

```
20160381@eelab13:~/ee412-bigdata/hw1$ date
Wed Oct  2 22:26:56 KST 2019
20160381@eelab13:~/ee412-bigdata/hw1$ python hw1_3.py articles.txt
t448    t8535
t8413   t269
t980    t2023
t1621   t7958
t3268   t7998
20160381@eelab13:~/ee412-bigdata/hw1$ date
Wed Oct  2 22:28:59 KST 2019
```