Q1a)

I divided this question into two map-reduce job.
The first map task is print a key-value pair similar to the input file.
The first reducer task print a key and a set, which the set will contain all the followee of that corresponding key.

The second mapper takes the first reducer output and generate every pair of possible combinations. If the length of their intersection is greater than 0, print it out.
The second reducer is just choose the greatest number of intersection of each blog id and print the information out.

1st MapReduce job:

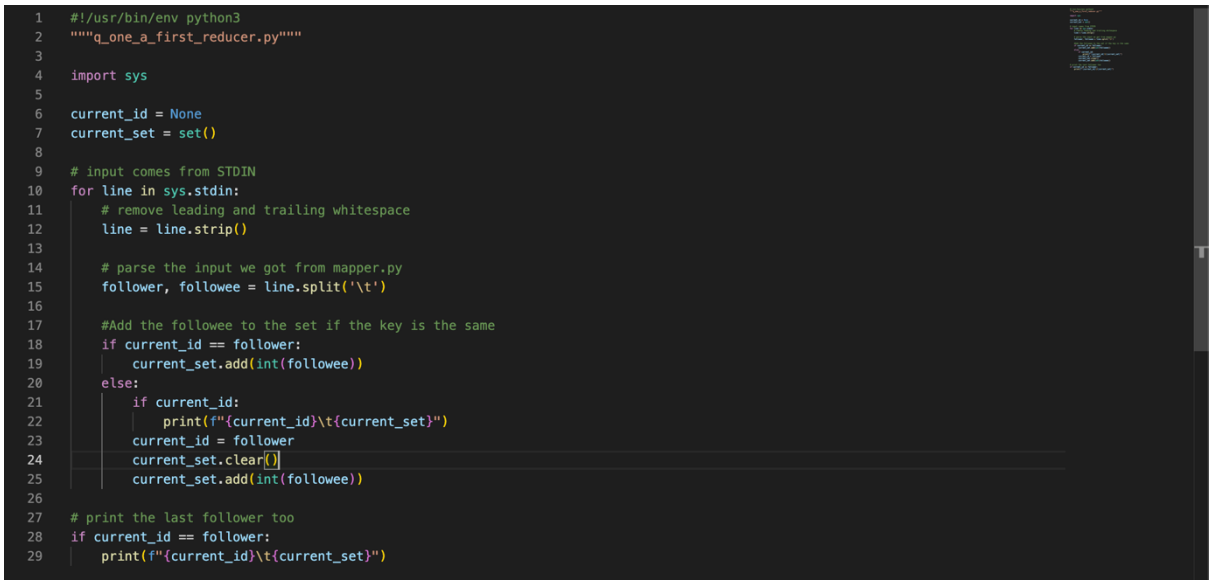
Command:
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -file ~/hw1/q_one_a_first_mapper.py -mapper ~/hw1/q_one_a_first_mapper.py -file ~/hw1/q_one_a_first_reducer.py -reducer ~/hw1/q_one_a_first_reducer.py -input medium/medium_relation -output q_one_a_1_output

```
[s1155157657@dicvmc4 hadoop]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -file ~/hw1/q_one_a_first_mapper.py  -mapper ~/hw1/q_one_a_first_mapper.py  -file ~/hw1/q_one_a_first_reducer
.py -reducer ~/hw1/q_one_a_first_reducer.py -input medium/medium_relation -output q_one_a_1_output
23/10/08 05:36:59 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/home/s1155157657/hw1/q_one_a_first_mapper.py, /home/s1155157657/hw1/q_one_a_first_reducer.py] [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar] /tmp/streamjob5619011106978l9953.jar
tmpDir=null
23/10/08 05:37:00 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 05:37:00 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 05:37:01 INFO mapred.FileInputFormat: Total input files to process : 1
23/10/08 05:37:01 INFO mapreduce.JobSubmitter: number of splits:2
23/10/08 05:37:01 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1694578679658_1036
23/10/08 05:37:01 INFO conf.Configuration: resource-types.xml not found
23/10/08 05:37:01 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/10/08 05:37:01 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
23/10/08 05:37:01 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
23/10/08 05:37:01 INFO impl.YarnClientImpl: Submitted application application_1694578679658_1036
23/10/08 05:37:01 INFO mapreduce.Job: The url to track the job: http://dicvmc3.ie.cuhk.edu.hk:8088/proxy/application_1694578679658_1036/
23/10/08 05:37:01 INFO mapreduce.Job: Running job: job_1694578679658_1036
23/10/08 05:37:06 INFO mapreduce.Job: Job job_1694578679658_1036 running in uber mode : false
23/10/08 05:37:06 INFO mapreduce.Job:  map 0% reduce 0%
23/10/08 05:37:12 INFO mapreduce.Job:  map 100% reduce 0%
23/10/08 05:37:19 INFO mapreduce.Job:  map 100% reduce 100%
23/10/08 05:37:19 INFO mapreduce.Job: Job job_1694578679658_1036 completed successfully
23/10/08 05:37:19 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=36003774
                FILE: Number of bytes written=72663317
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=32486915
                HDFS: Number of bytes written=18650436
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Rack-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=37756
                Total time spent by all reduces in occupied slots (ms)=36264
                Total time spent by all map tasks (ms)=9439
                Total time spent by all reduce tasks (ms)=4533
                Total vcore-milliseconds taken by all map tasks=9439
                Total vcore-milliseconds taken by all reduce tasks=4533
                Total megabyte-milliseconds taken by all map tasks=38662144
                Total megabyte-milliseconds taken by all reduce tasks=37134336
        Map-Reduce Framework
                Map input records=1768149
                Map output records=1768149
                Map output bytes=32467470
                Map output materialized bytes=36003780
                Input split bytes=252
                Combine input records=0
                Combine output records=0
                Reduce input groups=70097
                Reduce shuffle bytes=36003780
```

Code:
Mapper:

```python
#!/usr/bin/env python3
"""q_one_a_first_mapper.py"""

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split(" ")
    #print the pairs
    print('%s\t%s' % (words[0], words[1]))
```

Reducer:



```python
#!/usr/bin/env python3
"""q_one_a_first_reducer.py"""

import sys

current_id = None
current_set = set()

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    follower, followee = line.split('\t')

    #Add the followee to the set if the key is the same
    if current_id == follower:
        current_set.add(int(followee))
    else:
        if current_id:
            print(f"{current_id}\t{current_set}")
        current_id = follower
        current_set.clear()
        current_set.add(int(followee))

# print the last follower too
if current_id == follower:
    print(f"{current_id}\t{current_set}")
```

2nd MapReduce job:

Command:

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -D mapred.map.tasks=20 -D mapred.reduce.tasks=1 -files hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_a_1_output/part-00000 -file ~/hw1/q_one_a_second_mapper.py -mapper ~/hw1/q_one_a_second_mapper.py -file ~/hw1/q_one_a_second_reducer.py -reducer ~/hw1/q_one_a_second_reducer.py -input q_one_a_1_output/part-00000 -output q_one_a_two_16_output

```
[s1155157657@dicvmc4 ~]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -D  mapred.map.tasks=20 -D mapred.reduce.tasks=1 -files hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_a_1_output/part-00000 -file ~/hw1/q_one_a_second_mapper.py  -mapper ~/hw1/q_one_a_second_mapper.py  -file ~/hw1/q_one_a_second_reducer.py -reducer ~/hw1/q_one_a_second_reducer.py -input  q_one_a_1_output/part-00000 -output q_one_a_two_16_output
23/10/08 21:55:30 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/home/s1155157657/hw1/q_one_a_second_mapper.py, /home/s1155157657/hw1/q_one_a_second_reducer.py] [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar] /tmp/streamjob2998395375244070106.jar tmpDir=null
23/10/08 21:55:30 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 21:55:30 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 21:55:30 INFO mapred.FileInputFormat: Total input files to process : 1
23/10/08 21:55:30 INFO mapreduce.JobSubmitter: number of splits:20
23/10/08 21:55:31 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1694578679658_1435
23/10/08 21:55:31 INFO conf.Configuration: resource-types.xml not found
23/10/08 21:55:31 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/10/08 21:55:31 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
23/10/08 21:55:31 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
23/10/08 21:55:31 INFO impl.YarnClientImpl: Submitted application application_1694578679658_1435
23/10/08 21:55:31 INFO mapreduce.Job: The url to track the job: http://dicvmc3.ie.cuhk.edu.hk:8088/proxy/application_1694578679658_1435/
23/10/08 21:55:31 INFO mapreduce.Job: Running job: job_1694578679658_1435
23/10/08 21:55:37 INFO mapreduce.Job: Job job_1694578679658_1435 running in uber mode : false
23/10/08 21:55:37 INFO mapreduce.Job:  map 0% reduce 0%
23/10/08 21:55:45 INFO mapreduce.Job: Task Id : attempt_1694578679658_1435_m_000005_0, Status : FAILED
[2023-10-08 21:55:44.194]Container killed on request. Exit code is 137
[2023-10-08 21:55:44.194]Container exited with a non-zero exit code 137.
[2023-10-08 21:55:44.195]Killed by external signal

23/10/08 21:55:50 INFO mapreduce.Job:  map 2% reduce 0%
23/10/08 21:55:51 INFO mapreduce.Job:  map 3% reduce 0%
```

Code:

Mapper:

hw1 > q_one_a_second_mapper.py > ...

```python
#!/usr/bin/env python3
"""q_one_a_second_mapper.py"""

import sys
import ast

cache_lines = []

file_path = "part-00000"

# read the file from first mapreduce output
with open(file_path, 'r') as file:
    for line in file:
        # Do something with each line
        line = line.strip()
        temp_follower, temp_followee = line.split("\t",2)
        # turn the value to a set
        cache_lines.append((temp_follower, ast.literal_eval(temp_followee)))

# input comes from STDIN (standard input)
# for every line of input
for line in sys.stdin:
    max = 0
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    follower, followee_string = line.split("\t", 2)

    #process the value to a set
    followee_set = ast.literal_eval(followee_string)

    #loop throught the file read from first mapreduce output
    for cache_line in cache_lines:
        cache_follower, cache_followee_set = cache_line
        if cache_follower != follower:
            # find the intersection of the two sets
            intersection = followee_set.intersection(cache_followee_set)
            set_length = len(intersection)

            # if the length of the intersection is greater than 0
            if set_length >= max and set_length > 0:
                max = set_length
                # print the follower pairs, their intersection set and the length of the intersection
                print(f"{follower}\t{cache_follower}\t{intersection}\t{len(intersection)}")
```

Reducer:

```python
#!/usr/bin/env python3
"""q_one_a_second_reducer.py"""

import sys
import ast

current_website_one = None
current_website_two = None
current_followee_set = set()
current_followee_number = None
website_one = -1
# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    website_one, website_two, common_followee_string, common_followee_number = line.split('\t')

    # output the greatest length of a follower found
    if website_one != current_website_one:
        if current_website_one:
            print(f"{current_website_one}:{current_website_two}, {current_followee_set}, {current_followee_number}")

        current_website_one = website_one
        current_website_two = website_two
        current_followee_set.clear()
        current_followee_set = ast.literal_eval(common_followee_string)
        current_followee_number = len(current_followee_set)

    else:
        # check if the id is smaller when it is a tie in length/ length of set is longer
        if int(common_followee_number) > int(current_followee_number):
            current_website_one = website_one
            current_website_two = website_two
            current_followee_set.clear()
            current_followee_set = ast.literal_eval(common_followee_string)
            current_followee_number = len(current_followee_set)
        elif  int(common_followee_number) == int(current_followee_number) and int(current_website_two) > int(website_two):
            current_website_one = website_one
            current_website_two = website_two
            current_followee_set.clear()
            current_followee_set = ast.literal_eval(common_followee_string)
            current_followee_number = len(current_followee_set)

if current_website_one == website_one:
    print(f"{current_website_one}:{current_website_two}, {current_followee_set}, {current_followee_number}")
```

Final Output:
grep the blog ID ends with 7657 (my SID ends with 7657):

Command:
hadoop fs -cat q_one_a_two_16_output/part-00000 | grep “7657:”

```
[s11551576570dicvmc4 ~]$ hadoop fs -cat q_one_a_two_16_output/part-00000 | grep "7657:"
11576657:816394, {21347716, 72998223, 822388, 14296729, 14528658}, 5
15076657:18472741, {54795296, 14245665, 18744963, 25940370, 12946898, 19519258}, 6
1777076657:31209, {23030217, 11447858, 111691460, 71256938}, 4
1934676657:15693499, {14089201, 115485057, 18038275}, 3
4247876657:121569952, {240261248, 35701376, 16886231, 237420963, 284422694, 121773488, 94918923, 177345934, 153972655, 3359857, 110244437, 312199255, 18671608, 105643929, 116143610, 278109085}, 16
```

Q1b)

In this question I used the output from job1 in Q1a.
Based on the output of the 1st MapReduce job, we have a key and a set pair which the set contains all followee of the key. We only do one MapReduce job this time.

Mapper:
The mapper generates all possible pairs of keys (except with itself). If the length of their intersection of their corresponding set is greater than 0, then it output the similarity of this pair of keys and the intersection set (common followee).

```python
#!/usr/bin/env python3
"""q_one_b_first_mapper.py"""

import sys
import ast

cache_lines = []

file_path = "part-00000"

with open(file_path, 'r') as file:
    for line in file:
        # Do something with each line
        line = line.strip()
        temp_follower, temp_followee = line.split("\t",2)
        # make the value a set
        cache_lines.append((temp_follower, ast.literal_eval(temp_followee)))


# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    follower, followee_string = line.split("\t",2)
    followee_set = ast.literal_eval(followee_string)

    # find a pair of keys (self excluded)
    # if their length of intersection set is not 0, return the similarity and the intersection set
    for cache_line in cache_lines:
        cache_follower, cache_followee_set = cache_line
        if cache_follower != follower:
            intersection = followee_set.intersection(cache_followee_set)
            if len(intersection) != 0:
                union_len = len(followee_set) + len(cache_followee_set) - len(intersection)
                print(f"{follower}\t{cache_follower}\t{len(intersection)/union_len}\t{intersection}")
```

Reducer:
In the reduce stage I perform sorting for all the values for each key and take top K row of each key.

```python
#!/usr/bin/env python3
"""q_one_b_first_reducer.py"""

import sys
import ast

current_website_one = None
current_website_two = None
current_followee_set = set()
current_followee_number = None
top_k_list = []
k = 3
website_one = -1

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    website_one, website_two, similar_score, common_followee_string = line.split('\t')
    current_followee_set = ast.literal_eval(common_followee_string)

    # print top K result if key changed
    if current_website_one != website_one:
        if current_website_one:
            if len(top_k_list) > k:
                top_k_list = sorted(top_k_list, reverse=True, key=lambda x: (x[2]))
                for i in range(k):
                    print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")
            elif len(top_k_list) == k:
                for i in range(k):
                    print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")
            else:
                for i in range(len(top_k_list)):
                    print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")

        current_website_one = website_one
        top_k_list = []

    # add the information for the same key to set
    top_k_list.append((website_one,website_two,similar_score,current_followee_set))

# print top K result for the last one if key didn't changed
if current_website_one == website_one:
    if current_website_one:
        if len(top_k_list) > k:
            top_k_list = sorted(top_k_list, reverse=True, key=lambda x: (x[2]))
            for i in range(k):
                print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")
        elif len(top_k_list) == k:
            for i in range(k):
                print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")
        else:
            for i in range(len(top_k_list)):
                print(f"{top_k_list[i][0]}:{top_k_list[i][1]}, {top_k_list[i][3]}, {top_k_list[i][2]}")
```

MapReduce job:
Command:
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -D mapred.map.tasks=10 -D mapred.reduce.tasks=3 -files hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_a_1_output/part-00000 -file ~/hw1/q_one_b_first_mapper.py -mapper ~/hw1/q_one_b_first_mapper.py -file ~/hw1/q_one_b_first_reducer.py -reducer ~/hw1/q_one_b_first_reducer.py -input q_one_a_1_output/part-00000 -output q_one_b_5_output

```
[s1155157657@dicvmc4 hw1]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -D mapred.map.tasks=10 -D mapred.reduce.tasks=3 -files  hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_a_1_output/part-00000 -file ~/hw1/q_one_b_first_mapper.py  -mapper ~/hw1/q_one_b_first_mapper.py  -file ~/hw1/q_one_b_first_reducer.py -reducer ~/hw1/q_one_b_first_reducer.py -input  q_one_a_1_output/part-00000 -output q_one_b_5_output
23/10/08 23:18:40 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/home/s1155157657/hw1/q_one_b_first_mapper.py, /home/s1155157657/hw1/q_one_b_first_reducer.py] [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar] /tmp/streamjob8250016589354120990.jar tmpDir=null
23/10/08 23:18:40 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 23:18:40 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/08 23:18:41 INFO mapred.FileInputFormat: Total input files to process : 1
23/10/08 23:18:41 INFO mapreduce.JobSubmitter: number of splits:10
23/10/08 23:18:41 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1694578679658_1470
23/10/08 23:18:41 INFO conf.Configuration: resource-types.xml not found
23/10/08 23:18:41 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/10/08 23:18:41 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
23/10/08 23:18:41 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
23/10/08 23:18:41 INFO impl.YarnClientImpl: Submitted application application_1694578679658_1470
23/10/08 23:18:41 INFO mapreduce.Job: The url to track the job: http://dicvmc3.ie.cuhk.edu.hk:8088/proxy/application_1694578679658_1470/
23/10/08 23:18:41 INFO mapreduce.Job: Running job: job_1694578679658_1470
23/10/08 23:18:45 INFO mapreduce.Job: Job job_1694578679658_1470 running in uber mode : false
23/10/08 23:18:45 INFO mapreduce.Job:  map 0% reduce 0%
23/10/08 23:18:56 INFO mapreduce.Job:  map 1% reduce 0%
23/10/08 23:18:57 INFO mapreduce.Job:  map 3% reduce 0%
23/10/08 23:19:08 INFO mapreduce.Job:  map 4% reduce 0%
23/10/08 23:19:09 INFO mapreduce.Job:  map 5% reduce 0%
23/10/08 23:19:12 INFO mapreduce.Job:  map 6% reduce 0%
```

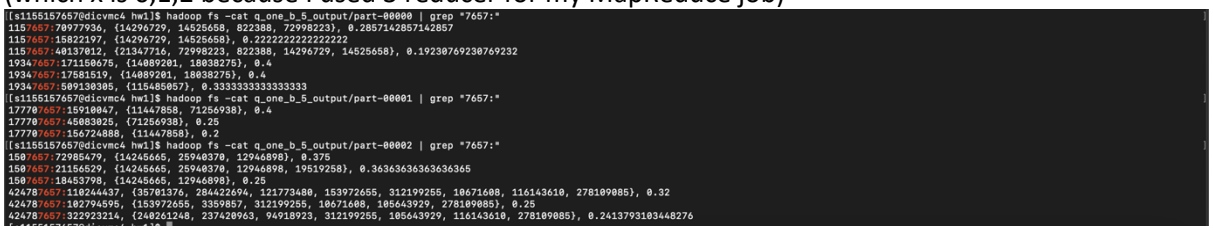
Output:

grep the blog ID ends with 7657 (my SID ends with 7657):

Command:

hadoop fs -cat q_one_b_3_output/part-0000x | grep "7657:"

(which x is 0,1,2 because I used 3 reducer for my MapReduce job)



Q1c)

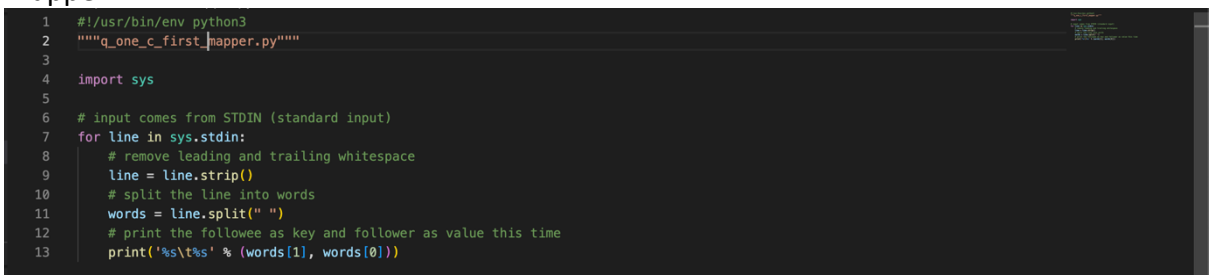
I used two MapReduce job in this question.
The first Mapper task is to take the medium_relation input and output the followee as key and the follower as value.
The first Reducer task is to output the blog id that has two or more follower.

Then, the second Mapper task is to take the output file from the first task and make it a list. Also take medium_label as input file. Read medium_label line by line and if the blog id is in the list, output a <community, 1> intermediate key if yes.
The second reducer simply sum it up for each community key and output the final value.

1st MapReduce job:

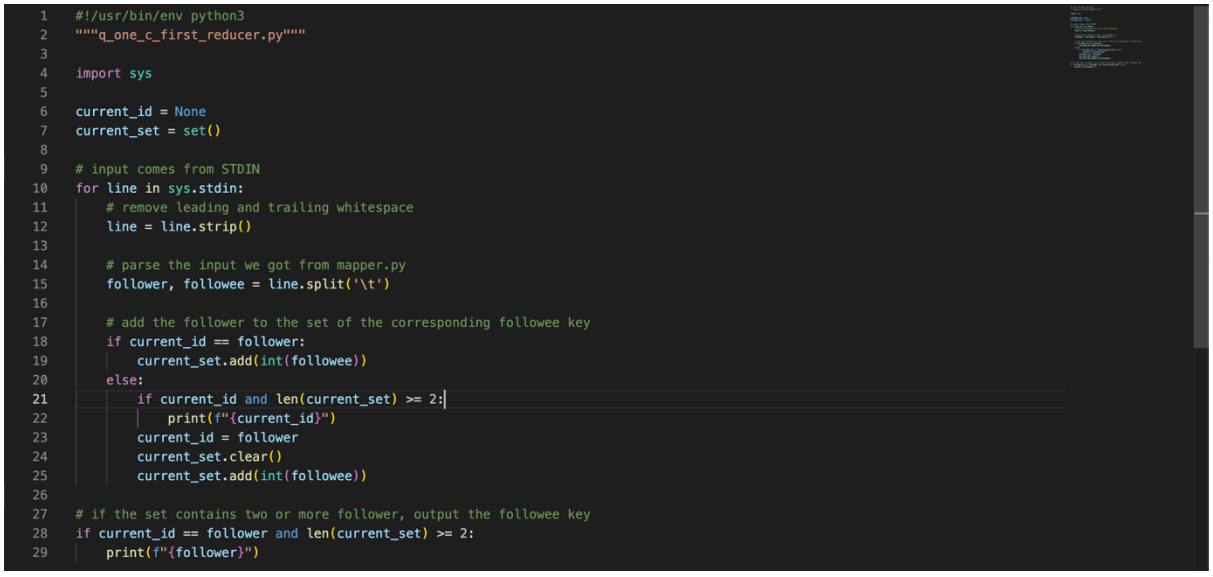
Code:

Mapper:



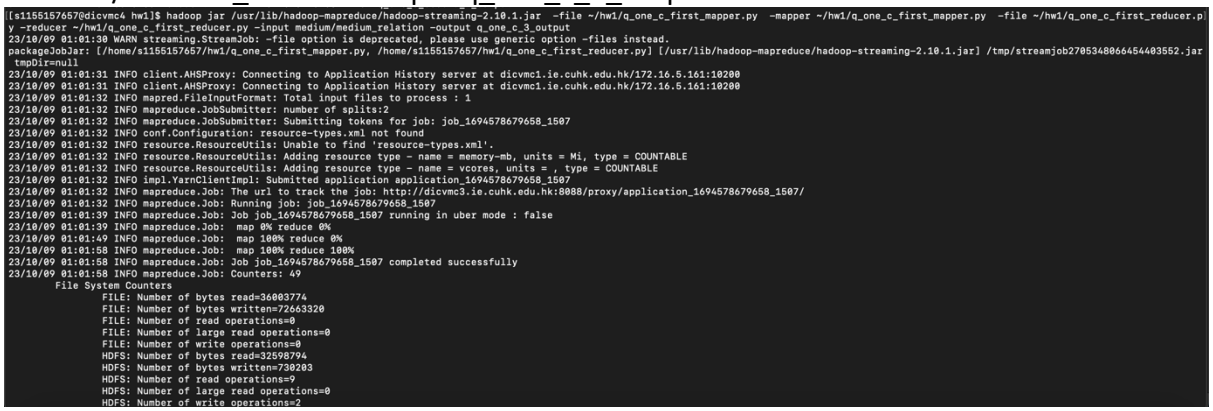
```python
#!/usr/bin/env python3
"""q_one_c_first_mapper.py"""

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split(" ")
    # print the followee as key and follower as value this time
    print('%s\t%s' % (words[1], words[0]))
```

Reducer:



```python
#!/usr/bin/env python3
"""q_one_c_first_reducer.py"""

import sys

current_id = None
current_set = set()

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    follower, followee = line.split('\t')

    # add the follower to the set of the corresponding followee key
    if current_id == follower:
        current_set.add(int(followee))
    else:
        if current_id and len(current_set) >= 2:
            print(f"{current_id}")
        current_id = follower
        current_set.clear()
        current_set.add(int(followee))

# if the set contains two or more follower, output the followee key
if current_id == follower and len(current_set) >= 2:
    print(f"{follower}")
```

Command:
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -file ~/hw1/q_one_c_first_mapper.py -mapper ~/hw1/q_one_c_first_mapper.py -file ~/hw1/q_one_c_first_reducer.py -reducer ~/hw1/q_one_c_first_reducer.py -input medium/medium_relation -output q_one_c_3_output



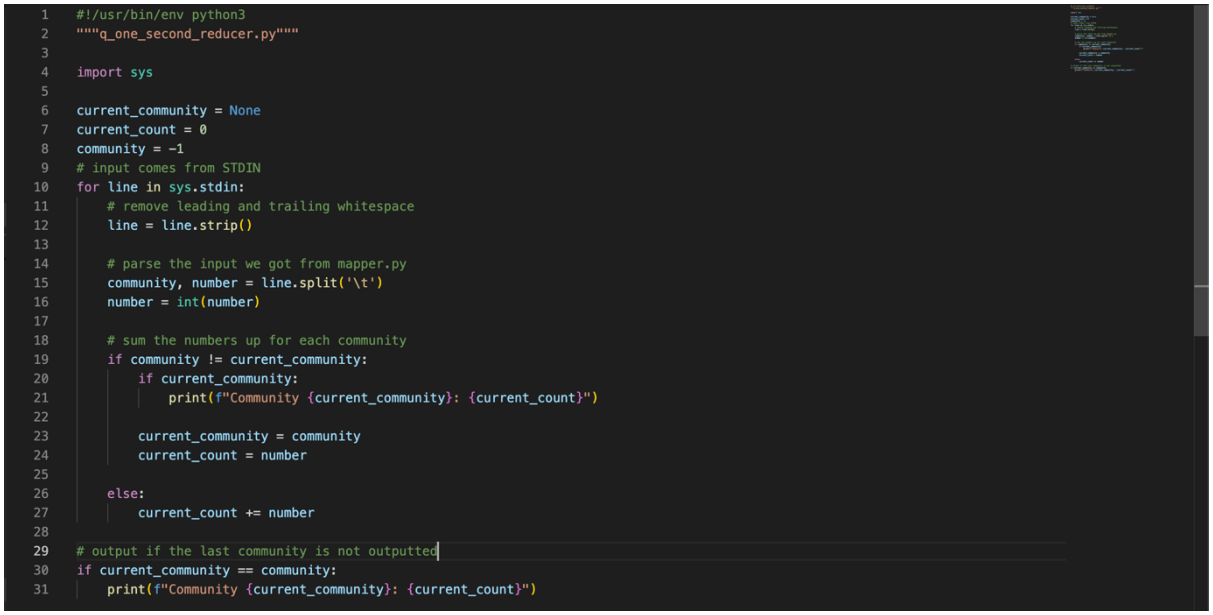
2nd MapReduce job:

Code:

Mapper:

```python
#!/usr/bin/env python3
"""q_one_c_second_mapper.py"""

import sys

cache_lines = []

file_path = "part-00000"

with open(file_path, 'r') as file:
    for line in file:
        # Do something with each line
        line = line.strip()
        cache_lines.append(int(line))

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    follower, community = line.split(" ", 2)
    # print(followee_set)

    # if the follower is in the first output file, the emit <community, 1>
    if int(follower) in cache_lines:
        print(f"{int(community)}\t1")
```

Reducer:



```python
#!/usr/bin/env python3
"""q_one_second_reducer.py"""

import sys

current_community = None
current_count = 0
community = -1
# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    community, number = line.split('\t')
    number = int(number)

    # sum the numbers up for each community
    if community != current_community:
        if current_community:
            print(f"Community {current_community}: {current_count}")

        current_community = community
        current_count = number

    else:
        current_count += number

# output if the last community is not outputted
if current_community == community:
    print(f"Community {current_community}: {current_count}")
```

Command:

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -files hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_c_3_output/part-00000 -file ~/hw1/q_one_c_second_mapper.py -mapper ~/hw1/q_one_c_second_mapper.py -file ~/hw1/q_one_c_second_reducer.py -reducer ~/hw1/q_one_c_second_reducer.py -input medium/medium_label -output q_one_c_second_2_output

```
[s1155157657@dicvmc4 hw1]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar  -files  hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_c_3_output/part-00000 -file ~/hw1/q_one_c_second_mapper.py  -mapper ~/hw1/q_one_c_second_mapper.py  -file ~/hw1/q_one_c_second_reducer.py -reducer ~/hw1/q_one_c_second_reducer.py -input medium/medium_label -output q_one_c_second_2_output
23/10/09 01:03:57 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/home/s1155157657/hw1/q_one_c_second_mapper.py, /home/s1155157657/hw1/q_one_c_second_reducer.py] [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar] /tmp/streamjob7544825735914073835.jar tmpDir=null
23/10/09 01:03:57 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/09 01:03:57 INFO client.AHSProxy: Connecting to Application History server at dicvmc1.ie.cuhk.edu.hk/172.16.5.161:10200
23/10/09 01:04:01 INFO mapred.FileInputFormat: Total input files to process : 1
23/10/09 01:04:01 INFO mapreduce.JobSubmitter: number of splits:2
23/10/09 01:04:01 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1694578679658_1509
23/10/09 01:04:02 INFO conf.Configuration: resource-types.xml not found
23/10/09 01:04:02 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/10/09 01:04:02 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
23/10/09 01:04:02 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
23/10/09 01:04:02 INFO impl.YarnClientImpl: Submitted application application_1694578679658_1509
23/10/09 01:04:02 INFO mapreduce.Job: The url to track the job: http://dicvmc3.ie.cuhk.edu.hk:8088/proxy/application_1694578679658_1509/
23/10/09 01:04:02 INFO mapreduce.Job: Running job: job_1694578679658_1509
23/10/09 01:04:11 INFO mapreduce.Job: Job job_1694578679658_1509 running in uber mode : false
23/10/09 01:04:11 INFO mapreduce.Job:  map 0% reduce 0%
23/10/09 01:04:21 INFO mapreduce.Job:  map 19% reduce 0%
23/10/09 01:04:22 INFO mapreduce.Job:  map 38% reduce 0%
23/10/09 01:04:24 INFO mapreduce.Job:  map 48% reduce 0%
23/10/09 01:04:25 INFO mapreduce.Job:  map 57% reduce 0%
23/10/09 01:04:27 INFO mapreduce.Job:  map 62% reduce 0%
23/10/09 01:04:31 INFO mapreduce.Job:  map 67% reduce 0%
23/10/09 01:04:33 INFO mapreduce.Job:  map 83% reduce 0%
23/10/09 01:04:34 INFO mapreduce.Job:  map 100% reduce 0%
23/10/09 01:04:38 INFO mapreduce.Job:  map 100% reduce 100%
23/10/09 01:04:38 INFO mapreduce.Job: Job job_1694578679658_1509 completed successfully
```

Output:

```
[s1155157657@dicvmc4 hw1]$ hadoop fs -cat q_one_c_second_2_output/part-00000
Community 0: 23633
Community 1: 23751
Community 2: 23705
```

Q1d)

#Job	Mapper num	Reducer num	Max mapper time	Min mapper time	Avg mapper time	Max reducer time	Min reducer time	Avg reducer time	Total time
1 (1036)	2	1	4s	4s	4s	2s	2s	2s	13s
2 (1041)	2	1	31m52s	29m46s	30m49s	9s	9s	9s	32m04s

#Job	Mapper num	Reducer num	Max mapper time	Min mapper time	Avg mapper time	Max reducer time	Min reducer time	Avg reducer time	Total time
1 (1562)	5	1	18s	3s	15s	3s	3s	3s	13s
2 (1584)	20	1	3m42s	2m58s	3m15s	17s	17s	17s	25m48s

#Job	Mapper num	Reducer num	Max mapper time	Min mapper time	Avg mapper time	Max reducer time	Min reducer time	Avg reducer time	Total time
1 (1596)	3	1	4s	4s	4s	2s	2s	2s	14s
2 (1598)	10	1	6m58s	5m56s	6m18s	16s	16s	16s	22m40s

#Job	Mapper num	Reducer num	Max mapper time	Min mapper time	Avg mapper time	Max reducer time	Min reducer time	Avg reducer time	Total time
1 (1614)	10	1	3s	3s	3s	2s	2s	2s	18s
2 (1615)	10	3	7m13s	6m1s	6m31s	4s	4s	4s	27m43s

Discovery:
For job1, since it is a very lightweight job, increasing amount of mapper does not help to boost the mapping stage but to slow it down on the whole progress.
For job2, since it requires a lot of system resources on mapping stage, so more mapper meaning more system resources can be utilized for mapper. It is generally faster if we use more mapper on mapping stage. However, is does not necessarily guarantee the whole job is

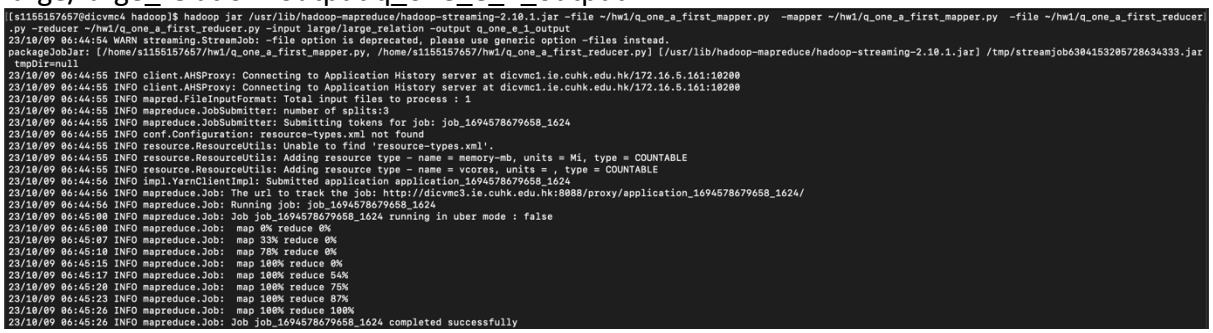
faster since it requires more time to shuffle and sort the result from more mappers. It slows down a bit when the mapper is too much.
Increasing reducer does not help in this case too because it is not reducer heavy job. No big performance boost is observed.

Bonus

I tried to run my code but it cannot work on the IE server. The first MapReduce job run successfully an it generates a key value pair where the key is the follower and the value is a set of followee of that follower.

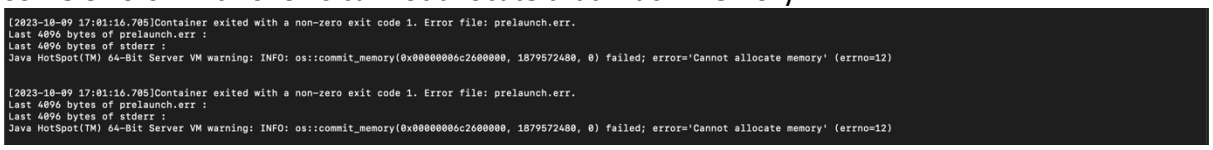
The first Job runs fine but the second one is messed up. The second job takes up too much memory

Command:
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -file ~/hw1/q_one_a_first_mapper.py -mapper ~/hw1/q_one_a_first_mapper.py -file ~/hw1/q_one_a_first_reducer.py -reducer ~/hw1/q_one_a_first_reducer.py -input large/large_relation -output q_one_e_1_output



hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.1.jar -D mapred.map.tasks=15 -D mapreduce.map.output.compress=true -files hdfs://dicvmc2.ie.cuhk.edu.hk/user/s1155157657/q_one_e_1_output/part-00000 -file ~/hw1/q_one_e_first_mapper.py -mapper ~/hw1/q_one_e_first_mapper.py -file ~/hw1/q_one_e_first_reducer.py -reducer ~/hw1/q_one_e_first_reducer.py -input q_one_e_1_output/part-00000 -output q_one_e_second_12_output

some errors which shows cannot allocate that much memory:



(Please give me some points for at least trying^^)

Q2a)

1)
Port 9000 is responsible for metadata services. It provides communication between datanode and resource manager. It is where the hdfs are connected to.
Port 8088 is yarn resource manager. It facilitates the execution of tasks in a job using existing resources available.

2)
I am using private IP. Using private IP is a saver than using public IP as public IP is exposed to security threads.

3)
No, as stated in the assignment we kept the internal communication port in gcp and it will allow communication of all machine in the same network. No extra firewall rules need to be setup.

Q2b)
In general, take up 150GB of total space in Hadoop using 4 VMs with 100GB storage each is feasible since 150GB < 400GB. However, we should consider other factors.

If the storage in the VMs cannot be fully utilized, such as they already have some big files that takes up bunch of storage. In that case, the total storage of hdfs may be less than 150GB. So, it does not have resources to store the 150GB in total in hdfs.

Also if you are refereeing to 150GB before storing it to hdfs, since by default it will have 3 replication and be stored to different VMs. So In this case it does not have enough storage for 150GB*3 = 450GB to store it.