COMP9313 Assignment01

zID: z5224151

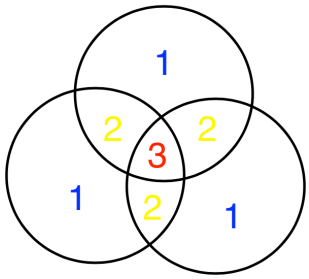
Name: ZANNING WANG

Q1:

1.

To take replication factor is 3 as an example, due to different blocks have same replicas, we can get the following Venn diagram. From the graphs, we can see that if one circle represent one block lost, number 1 in the figure means the data have lost one replicas, number 2 in the figure means the data have lost two replicas, number 3 in the figure means the data have lost three replicas. So that we can get L1(k,N) under the replication factor = 3 as below:

$$L1(k,N) = B \cdot k - 2 \cdot L2(k,N) - 3 \cdot L3(k,N)$$



This similar to the replication factor = 5, the L1(k,N) shows as below:

$$L1(k,N) = B \cdot k - 2 \cdot L2(k,N) - 3 \cdot L3(k,N) - 4 \cdot L4(k,N) - 5 \cdot L5(k,N)$$

In order to calculate the L2(k,N), we should calculate the number of lost-replicas when the k-th block broken firstly. And due to the replication factor is 4, we know that the number of block who lose one replicas equal to 4L1(k-1,N), therefore we can get when the k(th) block lost, there are $\frac{4 \cdot L1(k-1,N)}{N-(k-1)}$ blocks lost 2 replicas, and we should plus when the k-1(th) block lost, there are $L2(k-1,N)$ blocks, which lost 2 replicas, however, there are some part been mis-calculated, when the k(th) block lost, there are some block have already lost two replicas, we should minus it: $\frac{3 \cdot L2(k-1,N)}{N-(k-1)}$, therefore we can get the following formula:

$$L2(k,N) = \frac{4 \cdot L1(k-1,N)}{N-(k-1)} + L2(k-1,N) - \frac{3 \cdot L2(k-1,N)}{N-(k-1)}$$

The same as L3(k,N), L4(k,N), L5(k,N), the difference is coefficient is changed based on the number of replicas, therefore we can get the formulas like below:

$$L3(k,N) = \frac{3 \cdot L1(k-1,N)}{N-(k-1)} + L3(k-1,N) - \frac{2 \cdot L3(k-1,N)}{N-(k-1)}$$

$$L4(k,N) = \frac{2 \cdot L1(k-1,N)}{N-(k-1)} + L4(k-1,N) - \frac{1 \cdot L4(k-1,N)}{N-(k-1)}$$

$$L5(k,N) = \frac{1 \cdot L1(k-1,N)}{N-(k-1)} + L5(k-1,N) - \frac{0 \cdot L2(k-1,N)}{N-(k-1)}$$
$$= \frac{1 \cdot L1(k-1,N)}{N-(k-1)} + L5(k-1,N)$$

2.

This problem can be treated as dynamic planning, which means if we want to calculate the L5(2,500), we should ues the L4(1,500) and L5(1,500), the similar for L4(2,500), L3(2,500), L2(2,500),like below:

L4(1,500), L5(1,500) => L5(2,500)
L3(1,500), L4(1,500) => L4(2,500)
L2(1,500), L3(1,500) => L4(2,500)
L1(1,500), L2(1,500) => L2(2,500)

And L1(1,500) = B, L2(1,500) = L3(1,500) = L4(1,500) = L5(1,500) = 0

According to the result we get from the last questions, L1(2,500) can be calculated from L5(2,500), L4(2,500), L3(2,500), L2(2,500)., like below:

L5(2,500), L4(2,500), L3(2,500), L2(2,500) =>L1(2,500)

In this way, we can write code to calculate L5(200,500) = 39736.77280169178

```
/Users/zanning/opt/anaconda3/envs/COMP9313/bin/python
L1(200,500) = 1036781.3821642093
L2(200,500) = 1389356.8690281338
L3(200,500) = 923129.7317703705
L4(200,500) = 304107.9551149882
L5(200,500) = 39736.77280169178

Process finished with exit code 0
```

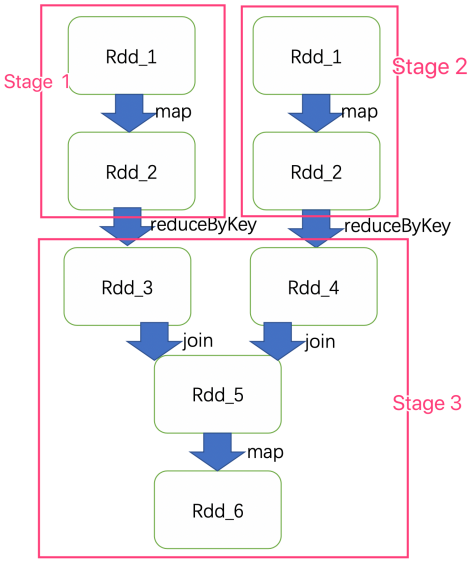
Q2:

1.

(1) In the rdd_2, we get the name with score: [name,score];
(2) In the rdd_3, we reduceByKey the rdd_2 and get the student with the highest score he get. [name, highest_score]
(3) In the rdd_4, we reduceByKey the rdd_2 and get the student with the lowest score he get. [name, lowest_score]
(4) In the rdd_5, we join rdd_3 and rdd_4 together. [name,highest_score,lowest_score]
(5) In the rdd_6, add highest score and the lowest score of each student. [name, total_score]
(6) Therefore we get the results as below:
[(Tina,155),(Jimmy, 159),(Thomas,167),(Joseph,165)]

2.

The stage are divided according the transformation is narrow transformation or wide transformation. Only after a wide transformation, there will be a new stage. Therefore in this questions, we got three stage



3.

According the code snippet given, there are two shuffle between rdd_3 and rdd_5, rdd_4 and rdd_5, which will decrease the efficiency of the spark, we can group the key by student with all the score they get, and sum the highest and the lowest in one step, the code show below:

```
rdd_1 = sc.parallelize(raw_data)
rdd_2 = sc.parallelize(raw_data)
rdd_3 = rdd_2.groupByKey()
rdd_4 = rdd_3.mapValues(lambda e :min(e) + max(e))
rdd_4.collect()
```

Q3:

1.

According to the assumption, the $\cos(\theta(o, q)) \geqslant 0.9$, the θ must less than 25.842°

$Pr[h_i(o) = h_i(q)] = 1- \theta/\pi = 1-0.144 = 0.856$

Therefore, we know $p^k_{q,o} > 0.856$

The probability of not finding any near duplicate is $(1 - p^k_{q,o})^l$

Therefore the equation is $1 – (1 - p^k_{q,o})^l \geqslant 0.99$, whereas $l= 8$.

In conclusion, at least 8 tables require to ensure the near duplicate with probability more than 99%.

2.

According to the assumption, the $\cos(\theta(o, q)) < 0.8$, the θ must more than 36.87°

$Pr[h_i(o) = h_i(q)] = 1- \theta/\pi = 1-0.205 = 0.795$

Therefore, we know $p^k_{q,o} < 0.795$

The probability of not finding any near duplicate is $(1 - p^k_{q,o})^l$

Therefore the equation is $1 – (1 - p^k_{q,o})^l = 1-(1-0.795)^{10} = 0.9782$

In conclusion, the maximum value of the probability of o to become a false positive of query q is 97.82%.