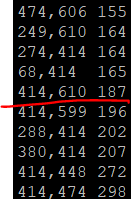


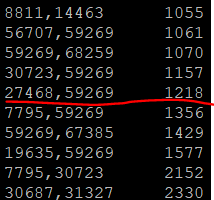
1155094482 Chim ka Long

**The overall result:**

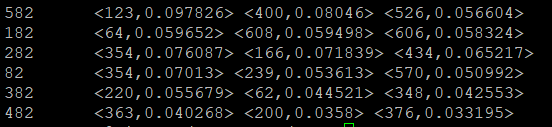
Part A small dataset:



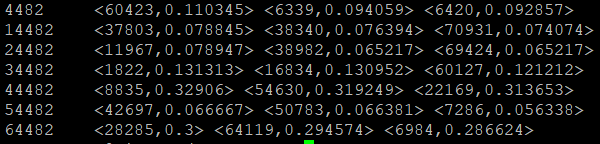
Part A large dataset:



Part B small dataset:



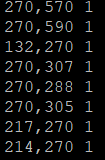
Part B large dataset:



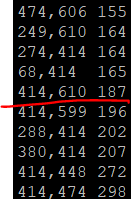
**Explanation**

Part A:

1. Map reduce 0: mapper output everything, but use movie id as key, reducer aggregate the users with same movie in array. Then process the array to emit pair of users if their ratings are same. The output like below

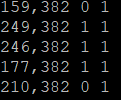


Map reduce 1: mapper output everything directly. Reducer sum the number and put into array. If array exceed certain number, it will do sorting and only hold the top 10 same rating. Finally, print them out

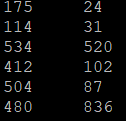


Part B:

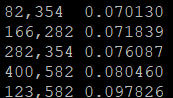
1. Map reduce 0: Different from part A, reducer only emit pair which include userid same as my last 2 digits. Also, it includes the number of same movies with same rating and the number of same movies regardless to rating. The output like below



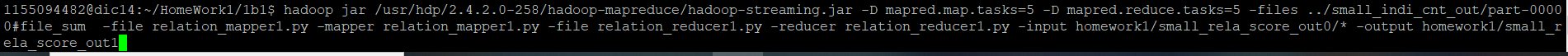
Map reduce Count: We calculate the sum of movies of every user.



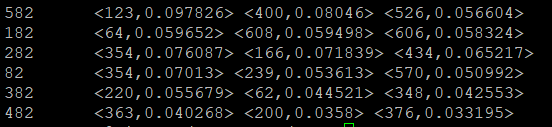
Map reduce 1: mapper only emit everything, and reducer will sum up the two numbers of same movies and calculate the similarity. Reducer need to read the sum of movies of every user firstly and according to user pair, calculate similarity = {no. of same movies and same rating}/{no. of movies of user A + no. of movies of user B – no. of same movies}. The output like below



The command like below

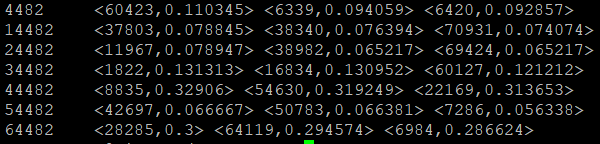


Map reduce 2: mapper filter the users of pair, if its last 2 digits is my cuid last 2 digits, emit it as Key and the partner userid + similarity as Value. Reducer will only print out the top 3 similar users with their id and similarity. Output like below



1. Handle large data set is almost same, except we filter with last 4 digits.

Output:



Part C:

Our map reduce job has 2 parts

Map reduce 0:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Max. Mapper time | Min. Mapper time | Average.  Mapper time | Max. Reducer time | Min. Reducer time | Average.  Reducer time | Total job |
| 5 mappers  5 reducers | 8s | 7s | 7s | 1h 29mins | 1h 1mins | 1h 14mins | 1h 41mins |
| 10 mappers  10 reducers | 13s | 5s | 11s | 59mins 55s | 25mins 9s | 43mins 25s | 1h 6s |
| 20 mappers  20 reducers | 7mins 5s | 4s | 6s | 42mins 4s | 11mins 29s | 21mins 21s | 42mins 18s |

Map reduce 1:

Although I only assign 5 mappers, the input file is too large, it is separated into many map tasks.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Max. Mapper time | Min. Mapper time | Average.  Mapper time | Max. Reducer time | Min. Reducer time | Average.  Reducer time | Total job |
| 5 mappers  5 reducers | 19mins 39s | 13s | 56s | 2h 34mins | 1h 46mins | 1h 2mins | 2h 44mins |
| 10 mappers  10 reducers | 9mins 15sec | 14s | 48s | 1h 22mins | 57mins 9s | 25mins 24s | 1h 35mins |
| 20 mappers  20 reducers | 2mins 2s | 6s | 46s | 57mins 16s | 24mins 36s | 12mins 31s | 1h 6mins |

Obviously, more mappers and reducers can make entire job faster. But it is not linearly. For example, if the mappers and reducers increase become 2 times, the total job elapsed time will not become half. There are mappers or reducers which completing job slowly, because their hardware is worse than others, or their jobs is much difficult. For example, in map reducer1, if the movie was watched by many users, the number of pairs emitted will exponentially increase. The job will become more difficult.

Reference for part C:

