Homework 2 – Finding Frequent Itemsets using SON Algorithm

SON Algorithm using Apriori:

First Map Phase:

First Read the data file. Using map function, we group by the keys. Map the values and pass it to mappartition() function for parallel processing. We use Apriori Algorithm here. First, create frequent candidate itemsets of size 1. If the total count of each item in the all the basket is greater than or equal to support (s) , it is a frequent item. The support threshold here is (support * (float (len(baskets_in_partition))/ float(total_basket_length))). The ratio is given for accurate results. We generate frequent items in phase 1. We have K phases in the algorithm based on itemset size. Itemsets can be frequent in phase K only if the subsets of itemsets are a candidate frequent in K-1 phase. In such way we create candidates frequent itemsets. The output is mapping key-value pairs (F,1). F is frequent items.

First Reduce Phase:

Using reducebykey(), this phase uses the result of map phase 1 and outputs candidate itemset which has count of one or more

Second Map Phase:

This takes all the candidate itemsets from first map-reduce phase and I check if each frequent item is subset of each basket. If yes, increase the value of item in dictionary by 1. This outputs (C,v) where c is candidate sets and v is support for that item in the baskets assigned to this particular map phase.

Second Reduce Phase:

If the count of itemset is greater than or equal to the given support, we consider those as frequent items.

Task 1:

Case 1 : Compute frequent businesses above a given support.

Case 2: Compute frequent users above a given support

Output in Lexicographical order with each size of itemset in different line.

Execution Timeline:

Input	Case	Support	Runtime (Sec)
small2.csv	1	4	9 secs <= 200
small2.csv	2	9	7 secs <= 100

Task 2:

1) Data-Preprocessing : State = “NV”
2) Get “user_id” and “business_id” from review json file whose “business_id” is from ”NV”.
3) Based on Filter Threshold and support generate frequent business sets

Execution Timeline:

Input	Filter Thresh	Support	Runtime (Sec)
user_business.csv	70	50	111 sec <=1500

Result Findings :

Consider a frequent triplet from the output file :

('4k3RlMAMd46DZ_JyZU0lMg', '7sPNbCx7vGAaH7SbNPZ6oA', 'JyxHvtj-syke7m9rbza7mA')
By analyzing, I found that, all of them belong to city Las Vegas. They all have a common category of restaurants. All of their ambience are “casual”. They all belong to star rating (3.5-4). Even the names of business are something on food. Eg. Ramen Sora, Bachi Burger, Sushi House. Thus all of these are authentic restaurants with special cuisine. The restaurant timings are quite similar from 11 am approx. to 12 pm.

Consider other frequent triplet from the output file:

('IMLrj2klosTFvPRLv56cng', 'igHYkXZMLAc9UdV5VnR_AA', 'qqs7LP4TXAoOrSlaKRfz3A')

They belong to stars rating of 4-4.5. All of them has ambience of “Trendy”. All of them have a price range of 2. Hence expensive. They all have high count of reviews above 1k. None of them provide breakfasts in their hotel.

Thus these are the few finding of frequent itemsets.