# Report

107062546 楊仲愷

**Java File, Class Name: FrequentItemsets01, FrequentItemsets**
**FrequentItemsets02, FrequentItemsets**
**FrequentItemsets03, FrequentItemsets**

**I implement frequent itemset algorithm and take advantage of a priori algorithm. Data are from the log of kkobx users.**

In jar FrequentItemsets01, first, I deal with the data from total2.txt, and I will get the processed data which are songs listened by users and output them and named output-01.txt because we can drop out the data which we don't use, and the size of file can be smaller.

```
[root@sandbox-hdp target]# yarn jar FrequentItemsets01.jar FrequentItemsets /Fin
alProject/Input /FinalProject/Output01
```

Total2.txt:

```
msno,song_id,source_system_tab,source_screen_name,source_type,target,song_l
DbQ3ldtGS3Nvw2mVFjmsOU6NxRu/it7haRG2q2AHtU8=,LKSUdsqrCOI+SnM0ccnf5DcxFYaAqc
/FprtuHWTmxwTrTEPX4Ux/6KMK2ULcKkGoj31BBWvtI=,Wg5sVFfhQsTNhdNcWwYEFIwOawBhoy
7NYpA3Wo2d0axLvHJ4HY14oeQV5adMJTbNXK3SJdJ9c=,kfqR9unVhcEyOgPEqc23jyuJa1ln45
cBT0mFe4knne5VgcBBUXMWU07N6W+XgtKEmvxFrTHYg=,G3Q5MWAPtj3MvQCzNMbs4tueDnPiXC
leTgUZpafVAhhcLxwH7cXbjAZN1dNt/ZDgbLWqI8nik=,ecY2Tb4ILEoX32uUNIk4dlGYwEq8Tu
BjZ+qxIhx0nS/OgUrelfGXyFXvwLsLRYQseZ730suv8=,yk5oMjkNyloHninWUwBJtZiYjtTv+u
```

output-01.txt:

```
黃乙玲,The Last Shadow Puppets
G.E.M.鄧紫棋
信 (Shin),Taeyeon,林宥嘉 (Yoga Lin),Suming
aMEI (張惠妹),林俊傑 (JJ Lin),羅志祥 (Show Lo)
孫盛希 (Shi Shi),Clean Bandit,Rick Ross,A-Lin,Justin Bieber,Best Pianc
邰正宵 (Samuel Tai)
楊丞琳 (Rainie Yang),郭采潔 (Amber Kuo)
張韶涵 (Angela Chang),Sia,陳奕迅 (Eason Chan),BJ The Chicago Kid,Vario
Space Cake 史貝絲考克,□□□□□□□□□□□,Various Artists,GOT7,Stacey Kent
范瑋琪 (Christine Fan),元衛覺醒 (Awaking),LEE HI
Imagine Dragons,徐譽滕,郁可唯 (Yisa Yu)
TWICE
```

Read total2.txt:

```
while (br.ready()) {
    ArrayList<String> temp = new ArrayList<String>();

    String tem[] = br.readLine().split(",");
    for(int i = 0; i < tem.length; i ++) {
        temp.add(tem[i]);
    }

    if(map.containsKey(temp.get(0))) {
        ArrayList<String> arr = new ArrayList<String>();
        arr = map.get(temp.get(0));

        String singer = temp.get(8);

        if(!arr.contains(singer)) {
            arr.add(singer);
        }

    }else {
        ArrayList<String> arr = new ArrayList<String>();
        String singer = temp.get(8);

        arr.add(singer);

        map.put(temp.get(0), arr);
    }
}
```

Write output-01.txt

```
for(Map.Entry<String, ArrayList<String>> entry : map.entrySet()) {
    ArrayList<String> arraylist1 = entry.getValue();
    String string = "";
    for(int i = 0; i < arraylist1.size(); i ++) {
        if(i < arraylist1.size() - 1) {
            string = string + arraylist1.get(i) + ",";
        }else {
            string = string + arraylist1.get(i);
        }

    }
    outputStream.write(string.getBytes(Charset.forName("UTF-8")));
    outputStream.writeBytes("\n");
}
```

Then, we count the number of songs which are listened. In mapper, we set the name of song with key and 1 with value.

```
public void map(Object key, Text values, Context context) th
    String singer[] = values.toString().split(",");

    Text word2 = new Text();
    word2.set("1");

    for(int i = 0; i < singer.length; i ++) {
        Text word1 = new Text();
        word1.set(singer[i]);

        context.write(word1, word2);
    }

}
```

In reducer, we count the number of song and only write songs which number are higher than 5.

```
public void reduce(Text key, Iterable<Text> values, Conte
    int count = 0;

    for (Text val : values) {
        count = count + Integer.parseInt(val.toString());
    }

    Text word2 = new Text(count + "");

    if(count >= 5) {
        context.write(key, word2);
    }
}
```

And we will get the output.

```
"M.A.D. (Lori Shen沈曉慧、Kevin Lee李冠毅、""POLO WL"")"  19
'N Sync 5
10cm    12
16個夏天 電視原聲帶       92
187 INC 謀殺有限公司      24
1976    7
2 Chainz| Wiz Khalifa   10
2006 KTV點唱精選國語總排行        19
2012網路票選權威歌曲      5
2012高中原創畢業歌合輯    5
2013網路票選權威歌曲      10
```

In jar FrequentItemsets02, we read the output from FrequentItemsets01, take advantage of a priori algorithm, and only write the data which are not deleted; then, write into freq2.txt. And count the number from freq2.txt.

```
[root@sandbox-hdp target]# yarn jar FrequentItemsets02.jar FrequentItemsets /Fin
alProject/Input /FinalProject/Output02
```

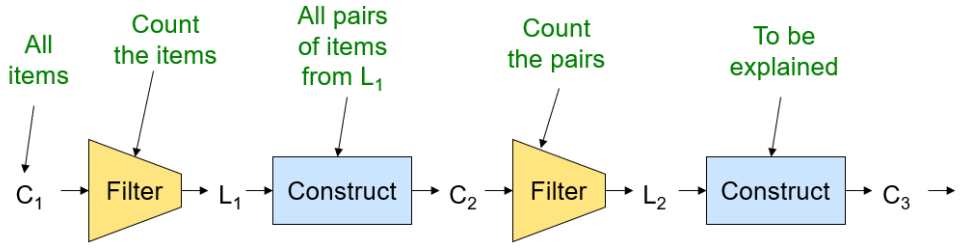
Read output01.txt

```
while (br1.ready()){
    String line[] = br1.readLine().split(",");
    ArrayList<String> array = new ArrayList<String>();

    for(int i = 0; i < line.length; i ++) {
        array.add(line[i]);
    }
    total.add(array);
}
```

Write the data which are not deleted:

```
for(int i = 0; i < total.size(); i ++) {
    ArrayList<String> arraylist1 = total.get(i);

    for(int j = 0; j < arraylist1.size() - 1; j ++) {
        String string1 = arraylist1.get(j);

        for(int k = j + 1; k < arraylist1.size(); k ++) {

            String string2 = arraylist1.get(k);

            if(fliter1.contains(string1) && fliter1.contains(string2)) {
                String string = string1 + "," + string2;

                outputStream1.write(string.getBytes(Charset.forName("UTF-8")));
                outputStream1.writeBytes("\n");
            }
        }
    }
}
```

A Priori Algorithm:




Pseudo-code:

$C_k$: Candidate itemset of size k
$L_k$ : frequent itemset of size k

$L_1$ = {frequent items};
**for** ($k$ = 1; $L_k$ !=∅; $k$++) **do begin**
  $C_{k+1}$ = candidates generated from $L_k$;
  **for each** transaction $t$ in database do
    increment the count of all candidates in $C_{k+1}$ that are contained in $t$
  $L_{k+1}$ = candidates in $C_{k+1}$ with min_support
  **end**
**return** $\cup_k L_k$;

freq2.txt:

```
黃乙玲,The Last Shadow Puppets
信 (Shin),Taeyeon
信 (Shin),林宥嘉 (Yoga Lin)
信 (Shin),Suming
Taeyeon,林宥嘉 (Yoga Lin)
Taeyeon,Suming
林宥嘉 (Yoga Lin),Suming
aMEI (張惠妹),林俊傑 (JJ Lin)
aMEI (張惠妹),羅志祥 (Show Lo)
林俊傑 (JJ Lin),羅志祥 (Show Lo)
孫盛希 (Shi Shi),Clean Bandit
孫盛希 (Shi Shi),A-Lin
孫盛希 (Shi Shi),Justin Bieber
```

And the mapper and reducer are like FrequentItemsets01. We will get the output.

```
16個夏天 電視原聲帶,Various Artists	8
16個夏天 電視原聲帶,周杰倫 (Jay Chou)	6
2PM,Various Artists	5
5566,Various Artists	6
5566,周杰倫 (Jay Chou)	5
A-Lin,Alan Walker	6
A-Lin,BIGBANG	6
A-Lin,BY2	7
A-Lin,BoA	6
A-Lin,Bruno Mars	7
A-Lin,CHARLIE PUTH	6
A-Lin,Coldplay	7
A-Lin,Ed Sheeran	5
A-Lin,Eric 周興哲	9
A-Lin,G.E.M.鄧紫棋	20
```

In jar FrequentItemsets03, we read the output from FrequentItemsets02, take advantage of a priori algorithm, and only write the data which are not deleted; then, write into freq3.txt. And count the number from freq3.txt.

Read the output from the original listened songs:

```
while (br1.ready()){
    String line[] = br1.readLine().split(",");
    ArrayList<String> array = new ArrayList<String>();

    for(int i = 0; i < line.length; i ++) {
        array.add(line[i]);
    }
    total.add(array);
}
```

Read the output from FrequentItemsets02:

```
while (br2.ready()){
    String line1[] = br2.readLine().split("\t");
    String line2[] = line1[0].split(",");

    ArrayList<String> array1 = new ArrayList<String>();

    for(int i = 0; i < line2.length; i ++) {
        array1.add(line2[i]);
    }

    filter2.add(array1);
}
```

freq3.txt:

```
aMEI (張惠妹),林俊傑 (JJ Lin),羅志祥 (Show Lo)
孫盛希 (Shi Shi),Clean Bandit,Various Artists
孫盛希 (Shi Shi),A-Lin,Justin Bieber
孫盛希 (Shi Shi),A-Lin,Various Artists
孫盛希 (Shi Shi),Justin Bieber,Various Artists
Clean Bandit,A-Lin,Various Artists
Clean Bandit,Justin Bieber,Various Artists
A-Lin,Justin Bieber,Various Artists
張韶涵 (Angela Chang),Sia,Various Artists
張韶涵 (Angela Chang),陳奕迅 (Eason Chan),Various Artists
張韶涵 (Angela Chang),Various Artists,Ed Sheeran
Sia,陳奕迅 (Eason Chan),Various Artists
Sia,陳奕迅 (Eason Chan),Ed Sheeran
```

Here, we take advantage of an algorithm which are from myself. I read every two-tuple set and if each of them is in three-tuple set and count every item in three tuple set and counts will be higher than two.

```
for(int i = 0; i < total.size(); i ++) {
    ArrayList<String> arraylist1 = total.get(i);

    for(int j = 0; j < arraylist1.size() - 2; j ++) {
        String string1 = arraylist1.get(j);

        for(int k = j + 1; k < arraylist1.size() - 1; k ++) {
            String string2 = arraylist1.get(k);

            for(int l = k + 1; l < arraylist1.size(); l ++) {
                ArrayList<String> temp = new ArrayList<String>();
                String string3 = arraylist1.get(l);

                temp.add(string1);
                temp.add(string2);
                temp.add(string3);

                if(countin(filter2, temp)) {
                    String string = string1 + "," + string2 + "," + string3;

                    outputStream1.write(string.getBytes(Charset.forName("UTF-8")));
                    outputStream1.writeBytes("\n");
                }
            }
        }
    }
}
```

output:

```
A-Lin,Various Artists,周杰倫 (Jay Chou)  5
A-Lin,五月天 (Mayday),周杰倫 (Jay Chou)  5
G.E.M.鄧紫棋,Various Artists,田馥甄 (Hebe)       5
Maroon 5,Various Artists,周杰倫 (Jay Chou)       7
Maroon 5,林俊傑 (JJ Lin),Various Artists 5
Various Artists,BIGBANG,五月天 (Mayday)  5
Various Artists,aMEI (張惠妹),G.E.M.鄧紫棋       5
Various Artists,aMEI (張惠妹),五月天 (Mayday)    6
```

In the final, I sort the output frequent itemset by their counts and get final.txt.

```
Comparator<Map.Entry<String, Integer>> valueComparator = new Comparator<Map.Entry<String,Integer>>() {
    @Override
    public int compare(Entry<String, Integer> o1,
            Entry<String, Integer> o2) {
        return o1.getValue()-o2.getValue();
    }
};

List<Map.Entry<String, Integer>> list = new ArrayList<Map.Entry<String,Integer>>(filter3.entrySet());

Collections.sort(list,valueComparator);

Path output2 = new Path(otherArgs[0] + "/final.txt");
FileSystem fout2 = FileSystem.get(new Configuration());
FSDataOutputStream outputStream2 = fout2.create(output2);

for (Map.Entry<String, Integer> entry : list) {
    String string = entry.getKey() + ":" + entry.getValue();

    outputStream2.write(string.getBytes(Charset.forName("UTF-8")));
    outputStream2.writeBytes("\n");
}
```

final.txt:

```
謝和弦 (R-chord),Various Artists,五月天 (Mayday):5
田馥甄 (Hebe),五月天 (Mayday),周杰倫 (Jay Chou):5
Various Artists,陳奕迅 (Eason Chan),林俊傑 (JJ Lin):5
Various Artists,田馥甄 (Hebe),周杰倫 (Jay Chou):5
Various Artists,五月天 (Mayday),蔡健雅 (Tanya Chua):5
五月天 (Mayday),周杰倫 (Jay Chou),張韶涵 (Angela Chang):5
陳奕迅 (Eason Chan),Various Artists,aMEI (張惠妹):5
aMEI (張惠妹),五月天 (Mayday),周杰倫 (Jay Chou):5
五月天 (Mayday),Various Artists,周杰倫 (Jay Chou):5
陳奕迅 (Eason Chan),五月天 (Mayday),aMEI (張惠妹):5
Various Artists,五月天 (Mayday),陳奕迅 (Eason Chan):5
```