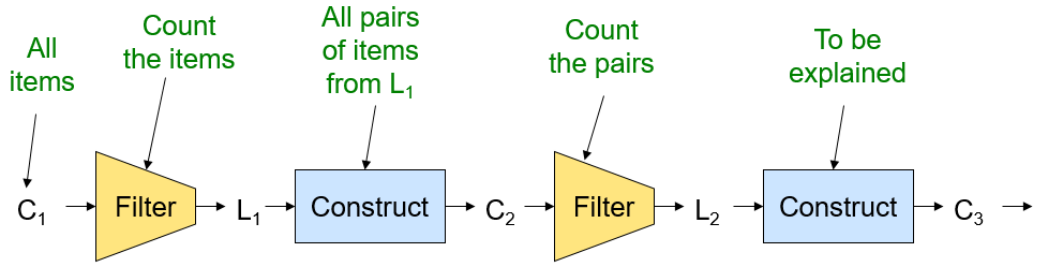
Report-implement the Apriori algorithm

第 21 組　108062468 張志宇

一、 Apriori 演算法介紹

- **For each *k*, we construct two sets of *k*-tuples** (sets of size *k*):
  - C_k = ***candidate k-tuples*** = those that might be frequent sets (support ≥ **s**) based on information from the pass for ***k*–1**
  - L_k = the set of truly frequent ***k***-tuples



1. 輸入 dataset 和 itemset，itemset 作爲初始候選集 C_1
2. 計算 C_1 中每個 item 的在 dataset 中的支持度（support），對比閾值（s）生成頻繁項集 L_1，此時 k=1
3. 將 $L_k \times L_k$,進行 cartesian product 得到候選集 C_{k+1}
4. 通過和 L_k 的比較，對 C_{k+1} 進行剪枝（prune）
5. 計算 C_{k+1} 中每個 item pair 的在 dataset 中的支持度（support），對比閾值（s）生成頻繁項集 L_{k+1}
6. 轉入步驟 3 直到符合給定的條件

二、 程式設計

1、第一個 mapreduce

Mapper 中[setup]讀取輸入的 itemset，把每一個 item 找出來；[map]中一行行

讀取 dataset，如果有某項 item，輸出: < key, value> = < item, 1>。

Reducer 中[setup] 讀取輸入的 minSupport 作爲判斷的閾值，[reduce]中計算 item 出現的次數，與 minSupport 比較，若大於等於則爲 frequent item。最後輸出：<key, value> = <NULL, frequent itemset >。

2、第二個 mapreduce

Mapper 中[setup]讀取前一次 iteration 的結果，然後執行 prune()進行 cartesian product 和 prune，產生這次要用的候選集 C_k。[map]中一行行讀取 dataset，如果有 C_k 中的 item pair，輸出: < key, value> = < item piar, 1>。

Reducer 中[setup] 讀取輸入的 minSupport 作爲判斷的閾值，[reduce]中計算 item pair 出現的次數，與 minSupport 比較，若大於等於則爲 frequent item。最後輸出：<key, value> = <NULL, frequent itemset >。

3、Command Line

arg[]= <dataset> <output> <itemset> <minSupport> <iterationNum>

三、 實驗

1、Small Dataset

Source：http://fimi.uantwerpen.be/data/中 retail (.gz)的前 50 行作爲 Dataset

retail.txt:

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32
33 34 35
36 37 38 39 40 41 42 43 44 45 46
38 39 47 48
38 39 48 49 50 51 52 53 54 55 56 57 58
32 41 59 60 61 62
3 39 48
63 64 65 66 67 68
32 69
48 70 71 72
39 73 74 75 76 77 78 79
36 38 39 41 48 79 80 81
82 83 84
41 85 86 87 88
39 48 89 90 91 92 93 94 95 96 97 98 99 100 101
36 38 39 48 89
39 41 102 103 104 105 106 107 108
```

ratailitem.txt(共 283 個 item):

```
0
1
10
100
101
102
103
104
105
106
107
108
109
11
110
111
112
```

設置 min support = 5, iteration number = 3, 結果如下：

L1:
```
32
36
38
39
41
48
```

L2:
```
36 38
36 39
38 39
38 48
39 41
39 48
41 48
```

L3:
```
36 38 39
38 39 48
39 41 48
```

2、Big Dataset

Source：http://fimi.uantwerpen.be/data/中的 chess (.gz)

chess.txt(共 3196 行，每行 37 個 item):

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 12 13 15 17 19 21 23 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 12 13 16 17 19 21 23 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 20 21 23 25 27 29 31 34 36 38 40 42 44 47 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 34 36 38 40 42 44 46 48 51 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 34 36 38 40 42 44 46 48 51 52 54 56 58 60 63 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 20 21 23 25 27 29 31 34 36 38 40 42 44 47 48 51 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 12 13 15 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 65 66 68 70 72 74
1 3 5 7 9 11 13 16 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 12 13 16 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 20 21 24 25 27 29 31 34 36 38 40 42 44 47 48 50 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 20 21 24 25 27 29 31 34 36 38 40 42 44 47 48 50 52 54 56 58 60 62 65 66 68 70 72 74
1 3 5 7 9 11 13 15 17 20 21 24 25 27 29 31 34 36 38 40 43 44 47 48 50 52 54 56 58 60 62 65 66 68 70 72 74
1 3 5 7 9 12 13 15 17 20 21 24 25 27 29 31 34 36 38 40 42 44 47 48 50 52 54 56 58 60 62 65 66 68 70 72 74
1 3 5 7 9 12 13 15 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 51 52 54 56 58 60 62 64 66 68 70 72 74
1 3 5 7 9 11 13 15 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 51 52 54 56 58 60 62 65 66 68 70 72 74
1 3 5 7 9 12 13 16 17 19 21 24 25 27 29 31 34 36 38 40 42 44 46 48 51 52 54 56 58 60 62 64 66 68 70 72 74
```

chessitem.txt(共 75 個 item):

```
1
10
11
12
13
14
15
16
17
18
19
2
20
21
22
23
24
25
```

由於 dataset 有 3000 多比資料， 所以我第一次實驗設置 min support = 1000，iteration number = 3, 結果如下：

L1: 具體結果見 S1000_L1.txt，共 47 個 frequent item

L2: 具體結果見 S1000_L2.txt，共 839 個 frequent item pair

L3: 具體結果見 S1000_L3.txt，共 8507 個 frequent item pair

可以看出，第一次實驗的結果 frequent item pair 的數量較多，因此進行第二次實驗，將 min support = 2000，iteration number = 3，結果如下：

L1: 具體結果見 S2000_L1.txt，共 31 個 frequent item

L2: 具體結果見 S2000_L2.txt，共 335 個 frequent item pair

L3: 具體結果見 S2000_L3.txt，共 1962 個 frequent item pair

可以看出，第二次實驗的結果 frequent item pair 的數量仍舊較多，因此進行第三次實驗，將 min support = 3000，iteration number = 3，結果如下：

```
29
34
36
40
48
52
56
58
60
62
66
7
```

L1: 共 12 個 frequent item

```
29 34
29 36
29 40
29 52
29 56
29 58
29 60
29 62
29 66
29 7
34 40
34 52
```

L2: 等共 38 個 frequent item pair，具體結果見 S3000_L2.txt

```
29 34 40
29 34 52
29 34 58
29 36 40
29 36 52
29 36 58
29 36 60
29 40 52
29 40 58
29 40 60
29 40 62
29 40 7
```

L3: 等共 55 個 frequent item pair，具體結果見 S3000_L3.txt

可以看出，第三次實驗的結果 frequent item pair 的數量已經不多，而這些 frequent item pair 的研究價值更高，透過這些 frequent item pair 的數位集合找出代表的 chess 可以進一步深入研究。