**Supermarket Optimization**

You’ve been hired as a market  consultant to try and help a local supermarket come up with better placement of items based on buyers preferences, and towards that goal you’d like to identify certain *association rules* based on existing records of buyers transactions.

You are given as input:

•   A transaction database - a file consisting of a single row per transaction, with individual product's SKUs given as space–separated integers. A single transaction consisting of products with SKUs 1001, 1002 and 1003 would have a line that looks like: ‘1001 1002 1003'

•   A minimal ’support level’ parameter , sigma – a positive integer

TO DO: Implement, in Python or Java,

an efficient algorithm **for generating all *frequent item sets*of size 3 or more**: NAMELY: **groups of 3 or more items** that appear **together** in the transactions log at **least as often as the support level parameter** value.

EXAMPLE: For example, given a value of sigma = 2, all sets of 3 items that appear 2 or more times together in the transaction log should be returned.

**The results should be returned as a file with the following format:**

<item *set size (N)>, <co-occurrence frequency>, <item 1 id >, <item 2 id>, …. <item N id>*

Run the algorithm on the attached transaction log file and provide the results obtained for a value of **sigma = 4**

Dear Adam,

I spent some time today doing a ‘close reading’ of the problem you sent. It is very fun! I have not yet started on the solution implementation as I would like to make sure 100% that I understand the statement of the problem correctly. I would like to note that this problem is reminiscent of the “Association rules mining A Priori algorithm” [I believe Agrawal 1993/1994?]. It is a famous study for supermarket basket cases. I read a long time ago this paper, but I never implemented it nor really though.

To ensure my correct understanding, let’s consider a very small case of the problem at hand. Let’s assume we have a db containing ONLY these rows: 1 – 6. Let’s further assume that our support level parameter sigma is 4 as in the original problem statement. From this small example we need to find ALL sets of 3 or MORE items that appear together at LEAST 4 times. Given our small example, and the statement of the problem above we will recover the following items I show in green.

Input data base reduced:

1) 38 39 41 105 110 487

2) 38 39 41 109 110

3) 38 39 41 48 60 367 368

4) 38 39 41 48 170 189

5) 36 38 39 41 48 79 80 81

6) 38 39 41 48 89 110

7) 39 48 592 593 594 595

8) 38 39 170 207 603

Output:

1. <4>, < co-occurance-frequency=4, <38> <39> <41> <48> *-> the rows we see this are 3, 4, 5, 6.*

2. <3>, <co-occurance-frequency=6>, <38> <39> <41> 🡺 *the rows we see this are 1-6 in data base.*

3. <3>, <co-occurance-frequency=4>, <38> <39> <48> 🡺 *the rows we see this are 3, 4, 5, 6 in data base.*

*Same for other subsets of cardinality 3 of the set of items: {*<38> <39> <41> <48> } e.g.

4. <3>, <co-occurance-frequency=4>, <38> <41> <48> 🡺 *the rows we see this are 3, 4, 5, 6 in data base.*

5. <3>, <co-occurance-frequency=4>, <39> <41> <48> 🡺 *the rows we see this are 3, 4, 5, 6 in data base.*

itemset frozenset(['39', '38', '48', '41']) Count 4

itemset frozenset(['39', '48', '41']) Count 4

itemset frozenset(['39', '38', '48']) Count 4

itemset frozenset(['38', '48', '41']) Count 4

itemset frozenset(['39', '38', '41']) Count 6

SIZE,SUPPORT,ITEMSET

3, 6, 38 39 41

3, 4, 38 39 48

3, 4, 38 41 48

3, 4, 39 41 48

4, 4, 38 39 41 48

Original – with no speed up:

--- 25318.4927561 seconds --- = total run time = 7hrs

With subset elimination at early stage -2 and 3 sets:

3476.16154218 seconds - = total run time = 57.936025703000006 minutes

Question:

1. Given that if items <38> <39> <41> <48> occur together it follows that <38> <39> <41> and their other subsets of size 3 also occur together. Do we list them as separate entries? Given the statement of the problem, my understanding would be that yes we want to list them as separate entries along the lines I show in output above. Please confirm though. One can imagine listing only the superset e.g. 4 items and not listing subsets of the requisite minimal size, here 3.
2. For the output format, would you prefer a csv file or .txt file with ‘,’ separating fields is acceptable? [happy to do csv]