CS 412: Fall ’23
Introduction To Data Mining

Assignment 5

(Due Monday, December 04, 2023, 11:59 pm)

- The homework is due at 11:59 pm on the due date. We will be using Gradescope for the homework assignments. If you are still having issues with joining Gradescope, you may use the code shared on a Canvas Announcement dated September 5. Please do NOT email a copy of your solution. Contact the TAs if you are having technical difficulties in submitting the assignment. We will NOT accept late submissions!

- Please use Slack or Canvas first if you have questions about the homework. You can also come to our (zoom) office hours and/or send us e-mails. If you are sending us emails with questions on the homework, please start subject with “CS 412 Fall’23: ” and send the email to *all of us* (Arindam, Ruby, Hyunsik, Rohan, Kowshika, Sayar) for faster response.

- Please write your code entirely by yourself.

  Programming Assignment Instructions

  – All programming needs to be in Python 3.

  – The homework will be graded using Gradescope. You will be able to submit your code as many times as you want. There will be two sequence databases: validation and test. Each will contribute half of the possible points on this problem. The validation questions will provide feedback but the test questions will not.

Assignment

The focus of the programming assignment is to develop code for a frequent sequential pattern mining algorithm based on PrefixSpan. Given a sequence database S and a minimum support threshold (minsup), the algorithm should return all the frequent sequential patterns. The sequential pattern is simplified to a **ordered sequential pattern**. In particular:

> All items in the sequences will be strictly ordered, i.e., there will be no sequences with co-occurring items. For example, there will be no sequence transactions of the form $\langle a(bc)d \rangle$ where (bc) has co-occurred.

The emphasis will be on correctness of the implementation, not memory efficiency. Since we will be testing the code on relatively small sequence databases S, you can assume all operations will be doable in main memory, and can use suitably data-structures to hold intermediate results. In particular, you do not have to create physical projected databases, which is needed when intermediate results cannot be held in main memory.

We will test your code on relatively small databases (maximum 20 sequences of length 20). Please make sure the runtime of your code does not exceed 10 seconds for such small databases.

You will not get any credit if your code does not work.

Input Format. The input will be a plain text file with a sequence database, with each line corresponding to a sequence. Each line will have a sequence id followed by the sequence. For example, for the sequence database given below:

Sequence_ID	Sequence
S_1	$\langle aabcacdcf \rangle$
S_2	$\langle adcbcae \rangle$
S_3	$\langle efabdfcb \rangle$
S_4	$\langle egafcbc \rangle$

the (validation) input text file will be:

```
s1, <aabcacdcf>
s2, <adcbcae>
s3, <efabdfcb>
s4, <egafcbc>
```

Your code will take two inputs:

1. A plain text file, the sequence database; and
2. An integer, the minimum support.

Output Format. Your code will implement a function called ord_prefixspan based on PrefixSpan, specialized to ordered sequential patterns, i.e., without having to consider co-occurring events. It will return a dictionary which will have keys as frequent patterns and values as the support of those patterns. The dictionary may look like:

$\{a : 4, ab : 4, \ ...\}$

What you have to submit

You need to submit a Python file named homework5.py. A starter code is posted on Canvas. Implement the rest to compute the required outputs. Your code should be in Python 3 and must include a function named ord_prefixspan which takes two inputs:

1. Sequence database (*filename* in the starter code): Name of a plain text file with the sequence database as shown in the example above. Each line will have a sequence Id and the sequence, and these two will be comma separated.
2. Minimum support (*minsup* in the starter code): An integer indicating the minimum support for the frequent pattern mining.

A call to the function will be like:
```
ord_prefixspan("seqDB.txt", 3)
```

The output, i.e., frequent ordered sequential patterns along with their support as shown in the example above, should be a dictionary.

Note: The submitted file needs to be named homework5.py, otherwise gradescope will give an error.

Additional Guidelines. The assignment needs you to both understand algorithms for sequential pattern mining, in particular PrefixSpan, as well as being able to implement the algorithm in python. Here are some guidelines to consider for the homework:

- You are getting about 3 weeks for the assignment, and many of you may need this time. If you are planning to start a week before the deadline, it is less likely you will be able to do a satisfactory job.
- It is good idea to make early progress on the assignment, so you can assess how much time it will take for you: (a) Start working on the assignment as soon as it is posted. Within the first week, you should have a sense of the parts which will be easier, and parts which will need extra effort from you; (b) Solve an example (partly) by hand as a warm-up to get comfortable with the steps which you will have to code. For the warm-up, you can use the example above or a suitably modified version of the example we discussed in class. The example in this outline is provided on Canvas in a text file named a5_sample_input.py.