Mining Periodic-Frequent Patterns in Temporal Databases

What is periodic-frequent pattern mining?

Periodic-Frequent pattern mining aims to discover all interesting patterns in a temporal database that have **support** no less than the user-specified **minimum support** (**minSup**) constraint and **periodicity** no greater than user-specified **maximum periodicity** (**maxPer**) constraint. The **minSup** controls the minimum number of transactions that a pattern must appear in a database and the **maxPer** controls the maximum time interval within which a pattern must reappear in the database.

Research paper: Tanbeer, Syed & Ahmed, Chowdhury & Jeong, Byeong-Soo. (2009). Discovering Periodic-Frequent Patterns in Transactional Databases. 5476. 242-253. 10.1007/978-3-642-01307-2_24 link.

What is a temporal database?

A temporal database is a collection of transactions at a particular timestamp, where each transaction contains a timestamp and a set of items.
A hypothetical temporal database containing the items ***a, b, c, d, e, f, and g*** as shown below

TS	Transactions
1	a b c g
2	b c d e
3	a b c d
4	a c d f
5	a b c d g
6	c d e f
7	a b c d
8	a e f
9	a b c d
10	b c d e

Note: Duplicate items must not exist in a transaction.

Acceptable format of temporal databases in PAMI

Each row in a temporal database must contain timestamp and items.

1 a b c g
2 b c d e
3 a b c d
4 a c d f
5 a b c d g
6 c d e f
7 a b c d
8 a e f
9 a b c d
10 b c d e

Understanding the statisctics of database

To understand about the database. The below code will give the detail about the transactional database.

- Total number of transactions (Database size)
- Total number of unique items in database
- Minimum lenth of transaction that existed in database
- Average length of all transactions that exists in database
- Maximum length of transaction that existed in database
- Minimum periodicity exists in database
- Average periodicity exists in database
- Maximum periodicity exists in database
- Standard deviation of transaction length
- Variance in transaction length
- Sparsity of database

The below sample code prints the statistical details of a database.

In [ ]:
```
import PAMI.extras.dbStats.temporalDatabaseStats as stats

obj = stats.temporalDatabaseStats('sampleInputFile.txt', ' ')
obj.run()
obj.printStats()
```

What is the input to periodic-frequent pattern mining algorithms

Algorithms to mine the periodic-frequent patterns requires temporal database, minSup and maxPer (specified by user).

- Temporal database is accepted following formats:
  - String : E.g., 'temporalDatabase.txt'
  - URL : E.g., https://u-aizu.ac.jp/~udayrage/datasets/transactionalDatabases/transactional_T10
  - DataFrame. Please note that dataframe must contain the header titled 'TS' and 'Transactions'
- minSup should be mentioned in
  - **count (beween 0 to length of database)**
  - [0, 1]
- maxPer should be mentioned in
  - **count (beween 0 to length of database)**
  - [0, 1]
- seperator
  default seperator is '\t' (tab space)

How to run the periodic-frequent pattern algorithm in terminal

- Download the PAMI source code from github.
- Unzip the PAMI source code folder and enter into periodic frequent pattern folder.
- Enter into frequentPattern folder
- You will find different types of folders like **basic, closed, maximal, topk**
- Enter into a specific folder of your choice and execute the following command on terminal.

syntax: python3 algorithmName.py `<path to the input file>` `<path to the output file>` `<minSup>` `<maxPer>` `<seperator>`

Example: python3 `PFPGrowth.py` `inputFile.txt` `outputFile.txt` `3` `4` `' '`

How to execute a periodic-frequent pattern mining algorithm in a Jupyter Notebook?

- Install the PAMI package from the PYPI repository by executing the following command: **pip3 install PAMI**
- Run the below sample code by making necessary changes

In [ ]:
```
import PAMI.periodicFrequentPattern.basic.PFPGrowth as alg

iFile = 'sampleInputFile.txt'  #specify the input transactional database
minSup = 5                     #specify the minSup value
maxPer = 3                     #specify the maxPer value
seperator = ' '                #specify the seperator. Default seperator
oFile = 'periodicPatterns.txt'   #specify the output file name

obj = alg.PFPGrowth(iFile, minSup, maxPer, seperator) #initialize the alg
obj.startMine()                        #start the mining process
obj.savePatterns(oFile)                #store the patterns in file
df = obj.getPatternsAsDataFrame()      #Get the patterns discovered into a
obj.printStats()                       #Print the statistics of mining pro
```

The periodicPatterns.txt file contains the following patterns (*format:* pattern:support:periodicity):!cat periodicPatterns.txt

In [3]:
```
!cat periodicPatterns.txt
```

```
a :7:2
a b :5:2
a b c :5:2
a d :5:3
a d c :5:3
a c :6:2
b :7:2
b d :6:2
b d c :6:2
b c :7:2
d :8:2
d c :8:2
c :9:2
```

The dataframe containing the patterns is shown below:

In [4]:
```
df
```

Out[4]:

	Patterns	Support	Periodicity
0	a	7	2
1	a b	5	2
2	a b c	5	2
3	a d	5	3
4	a d c	5	3
5	a c	6	2
6	b	7	2
7	b d	6	2
8	b d c	6	2
9	b c	7	2
10	d	8	2
11	d c	8	2
12	c	9	2