Mining Weighted Frequent Patterns in Uncertain Transactional Databases

What is weighted frequent pattern mining?

weighted Frequent pattern mining aims to discover all interesting patterns in a transactional database that have **support** no less than the user-specified **minimum support** (**minSup**) constraint and **weight** no less than the user-specified **minimum weight** (***minWeight***). The **minSup** controls the minimum number of transactions that a pattern must appear in a database. The **minWeight** controls the minimum weight of item.

What is the uncertain transactional database?

A transactional database is a collection of transactions, where each transaction contains a transaction-identifier and a set of items with ites repective uncertain value.
A hypothetical transactional database containing the items **_A, B, C, D, E, and F** as shown below

tid	Transactions
1	B(0.5) C(0.45) F(1.0)
2	A(0.7) B(0.82) D(0.3) F(0.75)
3	C(0.9) D(1.0) E(0.7)
4	A(0.48) B(0.8) C(0.6) D(1.0)
5	B(0.7) D(0.3) E(1.0)
6	B(0.65) C(1.0) D(0.8)
7	C(0.9) D(0.5) F(1.0)
8	A(0.4) E(0.4)
9	A(0.8) B(1.0) D(0.8) F(0.7)
10	B(0.4) C(0.9) D(1.0)

Note: Duplicate items must not exist in a transaction.

What is acceptable format of a transactional databases in PAMI

Each row in a transactional database must contain only items. The frequent pattern mining algorithms in PAMI implicitly assume the row number of a transaction as its transactional-identifier to reduce storage and processing costs. A sample transactional database, say sampleInputFile.txt, is provided below.

B(0.5) C(0.45) F(1.0)
A(0.7) B(0.82) D(0.3) F(0.75)
C(0.9) D(1.0) E(0.7)
A(0.48) B(0.8) C(0.6) D(1.0)
B(0.7) D(0.3) E(1.0)
B(0.65) C(1.0) D(0.8)
C(0.9) D(0.5) F(1.0)
A(0.4) E(0.4)
A(0.8) B(1.0) D(0.8) F(0.7)
B(0.4) C(0.9) D(1.0)

What is the Weighted database?

A weight database is a collection of items with their weights.
A hypothetical weight database containing the items ***A, B, C, D, E and F*** as shown below

A 0.40
B 0.70
C 1.00
D 0.55
E 0.85
F 0.30

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
- 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.transactionalDatabaseStats as stats
obj = stats.transactionalDatabaseStats('sampleInputFile.txt', ' ')
obj.run()
obj.printStats()
```

The input parameters to a frequent pattern mining algorithm are:

- **Transactional database**
  Acceptable formats:
  - String : E.g., 'transactionalDatabase.txt'
  - URL : E.g., https://u-aizu.ac.jp/~udayrage/datasets/transactionalDatabases/transactional_T10
  - DataFrame with the header titled 'Transactions'
- **minSup**
  specified in
  - **count (beween 0 to length of a database)** or
  - [0, 1]
- **minWeight**
  specified in
  - **count (beween 0 to length of a database)** or
  - [0, 1]
- **seperator**
  default seperator is '\t' (tab space)

How to store the output of a uncertain weighted frequent pattern mining algorithm?

The patterns discovered by a correlated pattern mining algorithm can be saved into a file or a data frame.

How to run the weighted frequent pattern mining algorithms in a terminal?

- Download the PAMI source code from github.
- Unzip the PAMI source code folder and enter into weighted uncertain frequent pattern folder.
- Enter into weightedUncertainFrequentPattern folder
- Enter into a specific folder and execute the following command on terminal.

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

Sample command to execute the WUFIM code in weightedUncertainFrequentPattern folder

Example: python3 `WUFIM.py` `inputFile.txt` `outputFile.txt` `weightSample.txt` `3` `2` `' '`

How to execute a weighted 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.weightedUncertainFrequentPattern.WUFIM as alg

iFile = 'sample.txt'  #specify the input transactional database <br>
wFile = 'HEWIWeightSample.txt'  #specify the input transactional database
minSup = 1.4  #specify the minSupvalue <br>
minWeight = 1.5    #specify the minWeight value <br>
seperator = ' ' #specify the seperator. Default seperator is tab space. <
oFile = 'weightedPatterns.txt'   #specify the output file name<br>

obj = alg.WUFIM(iFile, wFile, minSup, minWeight, seperator) #initialize t
obj.startMine()                         #start the mining process <br>
obj.savePatterns(oFile)                 #store the patterns in file <br>
df = obj.getPatternsAsDataFrame()       #Get the patterns discovered into a
obj.printStats()                        #Print the statistics of mining pro
```

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

In [2]:
```
!cat weightedPatterns.txt
```

```
E:2.1
C:4.75
C B:2.525
C B D:2.3
C D:3.65
B:4.870000000000001
B D:2.976
D:5.699999999999999
```

The dataframe containing the patterns is shown below:

In [3]:
```
df
```

Out[3]:

	Patterns	Support
0	(E,)	2.100
1	(C,)	4.750
2	(C, B)	2.525
3	(C, B, D)	2.300
4	(C, D)	3.650
5	(B,)	4.870
6	(B, D)	2.976
7	(D,)	5.700