Mining High-Utility Frequent Spatial Patterns in Spatiotemporal Utility Databases

What is High-Utility Frequent Spatial pattern mining?

High utility frequent spatial pattern mining aims to discover all the patterns with ***utility*** of pattern is no less than user-specified ***minimum utility*** (***minutil***), ***support*** of pattern is no less than ***minimum support*** (***minSup***) and the distance between any of its two items should not be more than user-specified ***maximum distace***.

What is the utility database?

A utility database is a collection of transaction, where each transaction contains a set of items and a positive integer called ***internal utility*** respectively. And each unique item in database is also associated with another positive number called ***external utility*** for each transaction.
A hypothetical utility database with items ***a, b, c, d, e, f and g*** and its ***internal utility*** is shown below at right side and items with its ***external utilities*** for each transaction is presented at left side.

Transactions	external utilities
(a,2) (b,3) (c,1) (g,1)	5 4 3 2
(b,3) (c,2) (d,3) (e,2)	5 2 9 3
(a,2) (b,1) (c,3) (d,4)	2 3 5 6
(a,3) (c,2) (d,1) (f,2)	1 3 4 6
(a,3) (b,1) (c,2) (d,1) (g,2)	2 5 3 6 1
(c,2) (d,2) (e,3) (f,1)	2 3 4 5
(a,2) (b,1) (c,1) (d,2)	5 4 3 2
(a,1) (e,2) (f,2)	4 8 3
(a,2) (b,2) (c,4) (d,2)	7 4 9 8
(b,3) (c,2) (d,2) (e,2)	5 9 10 24

Note: Duplicate items must not exist in a transaction.

Acceptable format of utility databases in PAMI

Each row in a utility database must contain only items, total sum of utilties and external utility values.

a b c g:7:2 3 1 1:5 4 3 2
b c d e:10:3 2 3 2:5 2 9 3
a b c d:10:2 1 3 4:2 3 5 6
a c d f:7:3 2 1 2:1 3 4 6
a b c d g:9:3 1 2 1 2:2 5 3 6 1
c d e f:8:2 2 3 1:2 3 4 5
a b c d:6:2 1 1 2:5 4 3 2
a e f:5:1 2 2:4 8 3
a b c d:10:2 2 4 2:7 4 9 8
b c d e:9:3 2 2 2:5 9 10 24

What is neighbour file?

A neighbour files consists item and set of its neighbours. A hypothetical neighbour database is defined below

Item	Neighbours
a	b, c, d
b	a, e, g
c	a, d
d	a, c
e	b, f
f	e, g
g	b, f

Accepted format of neighbour database in PAMI

a b c d
b a e g
c a d
d a c
e b f
f e g
g b f

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 utility value exists in database
- Average utility exists in database
- Maximum utility exists in database
- Standard deviation of transaction length
- Variance in transaction length
- Sparsity of database

The sample code

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

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

What are the input parameters?

Algorithms to mine the high-utility patterns requires utility database, minUtil (specified by user).

- Utility database can be provided in following formats:
  - String : E.g., 'utilityDatabase.txt'
  - URL : E.g., https://u-aizu.ac.jp/~udayrage/datasets/transactionalDatabases/transactional_T10
  - In DataFrame format (dataframe variable with heading `Transactions`, `Utilities` and `TransactionUtility`
- Neighbour database can be provided in following formats:
  - String : E.g., 'neighbourDatabase.txt'
- **minUtil**
  specified in
  - [0, 1]
- **minSup**
  specified in
  - **count (beween 0 to length of a database)** or
  - [0, 1]
- **seperator**
  default seperator is '\t' (tab space)

How to run the high-utility frequent spatial pattern algorithm in terminal

- Download the PAMI source code from github.
- Unzip the PAMI source code folder and enter into high utility frequent spatial pattern folder.
- Enter into highUtilityFrequentSpatialPattern folder
- You will find another folder **basic**
- Enter into a specific folder of your choice and execute the following command on terminal.

And execute the following command on terminal.

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

Example: python3 `SHUFIM.py` `inputFile.txt` `outputFile.txt` `neighbourFile.txt` `$20$` `$5$` `' '`

How to implement the SHUFIM algorithm by importing PAMI package

- 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.highUtilityFrequentSpatialPattern.basic.SHUFIM as alg

iFile = 'sampleUtility.txt'          #specify the input transactional dat
nFile = 'sampleNeighbourFile.txt'  #specify the input transactional datab
minUtil = 20                       #specify the minUtil value
minSup = 5                         #specify the minSup value
seperator = ' '                    #specify the seperator. Default sepera
oFile = 'utilityfrequentPatterns.txt'   #specify the output file name

obj = alg.SHUFIM(iFile, nFile, minUtil, minSup, seperator) #initialize th
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 utilityfrequentPatterns.txt file contains the following patterns (*format:* pattern:utility:support):!cat utilityfrequentPatterns.txt

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

```
c : 27:13
c d : 50:11
c d a : 49:7
c a : 40:9
d : 25:11
d a : 32:7
a : 22:10
a b : 27:7
b : 21:10
```

The dataframe containing the patterns is shown below:

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

Out[4]:

	Patterns	Utility:Support
0	c	27:13
1	c d	50:11
2	c d a	49:7
3	c a	40:9
4	d	25:11
5	d a	32:7
6	a	22:10
7	a b	27:7
8	b	21:10