Pollution

June 16, 2023

[1]:
```python
import pandas as pd
import Topk_PPPGrowth as tp
```

[2]:
```python
inputFile = 'Temporal_airpollutionJapan.csv'
seperator = '\t'
k = [200, 300, 400, 500, 600]
maxPer = 250

totalResult = pd.DataFrame(columns=['algorithm', 'minSup', 'maxPer','patterns',␣
 ↪'runtime', 'memory'])
#initialize a data frame to store the results of PFECLAT algorithm
```

[3]:
```python
algorithm = 'TOPK-3P'  #specify the algorithm name
for i in k:
    obj1 = tp.Topk_PPPGrowth(inputFile, k=i, periodicity=maxPer, sep=seperator)
    obj1.startMine()
    obj1.save('patterns.txt')
    #store the results in the data frame
    totalResult.loc[totalResult.shape[0]] = [algorithm, i, maxPer, len(obj1.
 ↪getPatterns()), obj1.getRuntime(), obj1.getMemoryRSS()]
```

```
200 200 250
TopK partial periodic patterns were generated successfully
300 300 250
TopK partial periodic patterns were generated successfully
400 400 250
TopK partial periodic patterns were generated successfully
500 500 250
TopK partial periodic patterns were generated successfully
600 600 250
TopK partial periodic patterns were generated successfully
```

[4]:
```python
print(totalResult)
```

```
  algorithm  minSup  maxPer  patterns    runtime     memory
0   TOPK-3P     200     250       200   5.738050  148946944
1   TOPK-3P     300     250       300  10.184110  150126592
2   TOPK-3P     400     250       400  16.491211  150978560
```

```
3   TOPK-3P     500     250       500  23.939137  151494656
4   TOPK-3P     600     250       600  33.095733  152051712
```

[5]:
```
def getTopPatterns(iFile, k):
    res = {}
    with open(iFile, 'r') as f:
        for line in f:
            line = line.split(':')
            res[line[0]] = line[1]
    res1 = {k:v for k, v in sorted(res.items(), key=lambda x:x[1],␣
 ↪reverse=True)}
    res1 = {k:v for k,v in list(res1.items())[:k]}
    return res1
```

[8]:
```
import time
import os as _os
import os.path as _ospath
import psutil as _psutil
from PAMI.partialPeriodicPattern.basic import PPPGrowth as pf
startTime = time.time()
for i in [200, 300, 400, 500, 600]:
    obj = pf.PPPGrowth(inputFile, 350, 250, '\t')
    obj.startMine()
    obj.save("patterns_t10.txt")
    patterns = getTopPatterns("patterns_t10.txt", i)
    endTime = time.time()
    memoryUSS = float()
    process = _psutil.Process(_os.getpid())
    memoryUSS = process.memory_full_info().uss
    print("Total Number of patterns:", len(patterns))
    print("Total Memory Taken:", memoryUSS)
    print("Total Time Taken:", endTime - startTime)
```

```
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 200
Total Memory Taken: 464441344
Total Time Taken: 69.37121152877808
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 300
Total Memory Taken: 445898752
Total Time Taken: 137.25756120681763
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 400
Total Memory Taken: 445878272
Total Time Taken: 204.7194790840149
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 500
Total Memory Taken: 444846080
Total Time Taken: 272.1290020942688
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 600
Total Memory Taken: 443801600
Total Time Taken: 339.4690887928009
```

[9]:
```python
from PAMI.extras.graph import generateLatexFileFromDataFrame as gdf
gdf.generateLatexCode(totalResult)
```

```
Latex files generated successfully
```

[ ]:
```python

```