T10

June 16, 2023

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

[2]:
```
inputFile = 'Temporal_T10I4D100K.csv'
seperator = '\t'
k = [500, 1000, 1500, 2000, 2500]
maxPer = 2000

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

[3]:
```
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()]
```

```
500 500 2000
TopK partial periodic patterns were generated successfully
868 1000 2000
TopK partial periodic patterns were generated successfully
868 1500 2000
TopK partial periodic patterns were generated successfully
868 2000 2000
TopK partial periodic patterns were generated successfully
868 2500 2000
TopK partial periodic patterns were generated successfully
```

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

```
  algorithm  minSup  maxPer  patterns    runtime     memory
0   TOPK-3P     500    2000       500   8.854007  223039488
1   TOPK-3P    1000    2000      1000  25.070870  223944704
2   TOPK-3P    1500    2000      1500  29.216477  224673792
```

```
3   TOPK-3P    2000    2000       2000  34.266527  224821248
4   TOPK-3P    2500    2000       2500  41.289999  224956416
```

[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
```

[6]:
```
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 [500, 1000, 1500, 2000, 2500]:
    obj = pf.PPPGrowth(inputFile, 100, 2000, '\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: 500
Total Memory Taken: 394031104
Total Time Taken: 13.839305877685547
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 1000
Total Memory Taken: 402038784
Total Time Taken: 27.2118239402771
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 1500
Total Memory Taken: 428318720
Total Time Taken: 40.16120719909668
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 2000
Total Memory Taken: 444043264
```

```
Total Time Taken: 53.68311095237732
Partial Periodic Patterns were generated successfully using 3PGrowth algorithm
Total Number of patterns: 2500
Total Memory Taken: 457633792
Total Time Taken: 67.1781370639801
```

[8]:
```
from PAMI.extras.graph import generateLatexFileFromDataFrame as gdf
gdf.generateLatexCode(totalResult)
```

```
Latex files generated successfully
```

[ ]:
```

```