# Discovering Partial Periodic Frequent Pattern in Big Data Using PPF_DFS Algorithm

In this tutorial, we will discuss two approaches to find Partial Periodic Frequent Pattern in big data using PPF_DFS algorithm.

1. **Basic approach:** Here, we present the steps to discover Partial Periodic Frequent Pattern using a single minimum support value
2. **Advanced approach:** Here, we generalize the basic approach by presenting the steps to discover Partial Periodic Frequent Pattern using multiple minimum support values.

---

## Basic approach: Executing PPF_DFS on a single dataset at a particular minimum support value

### Step 1: Import the PPF_DFS algorithm

```python
In [1]: from PAMI.partialPeriodicFrequentPattern.basic import PPF_DFS as alg
```

### Step 2: Specify the following input parameters

```python
In [2]: inputFile = 'temporal_T10I4D100K.csv'
maxPerCount=500
minPRcount=0.5
minimumSupportCount=100   #Users can also specify this constraint between 0 to 1.

seperator='¥t'
```

### Step 3: Execute the PPF_DFS algorithm

```python
In [3]: obj = alg.PPF_DFS(iFile=inputFile, minSup=minimumSupportCount,maxPer=maxPerCount,mi
obj.startMine()            #Start the mining process
```

### Step 4: Storing the generated patterns

### Step 4.1: Storing the generated patterns in a file

```python
In [4]: obj.savePatterns(outFile='frequentPatternsMinSupCount1000.txt')
```

### Step 4.2. Storing the generated patterns in a data frame

```python
In [5]: frequentPatternsDF= obj.getPatternsAsDataFrame()
```

### Step 5: Getting the statistics

### Step 5.1: Total number of discovered patterns

In [6]:
```python
print('Total No of patterns: ' + str(len(frequentPatternsDF)))
```

Total No of patterns: 20748

### Step 5.2: Runtime consumed by the mining algorithm

In [7]:
```python
print('Runtime: ' + str(obj.getRuntime()))
```

Runtime: 41.625649213790894

In [8]:
```python
##### Step 5.3: Total Memory consumed by the mining algorithm
```

In [9]:
```python
print('Memory (RSS): ' + str(obj.getMemoryRSS()))
print('Memory (USS): ' + str(obj.getMemoryUSS()))
```

Memory (RSS): 227917824
Memory (USS): 188776448