Student Name : Jaya Sistla
Course : CSI 5500 Operating Systems
Assignment 2: Implementing The Concept of Multithreading
ReadMe File

Requirement:

The data consists of set of transactions, each transaction consists of item Ids of the items purchased in a single transaction.

Use Case 1: We need to report the the item ID of two items which are most frequently purchased together and the probability of purchasing Item A given that item B is purchased.

Use Case 2: We need to report the item ID of three items which are most frequently purchased together and probability of purchasing Item A given that items B and C are purchased.

Design: The Transaction Data has 88162 number of lines and we need to create transaction pairs for every item with other item in the same transaction.
I.e 0 1 2 3 4 -> (0,1) (0,2) (0,3) (0,4) (2,3) (2,4) (3,4) are the possible number of pairs.
To implement this for such huge dataset I have used the concept of Multi threading to form pairs of items in each transaction.

Implementation:

I have used Python programming language for solving this problem. Python is well known for fast data processing and analysis capabilities, which motivated me to implement this program using multi threading.

Libraries Used : I have used a library called threading to implement multithreading.
To implement multi threading using “threading” library we need to define a class which initialises and runs a thread. Each thread is an object of this Class.

The class has two functions

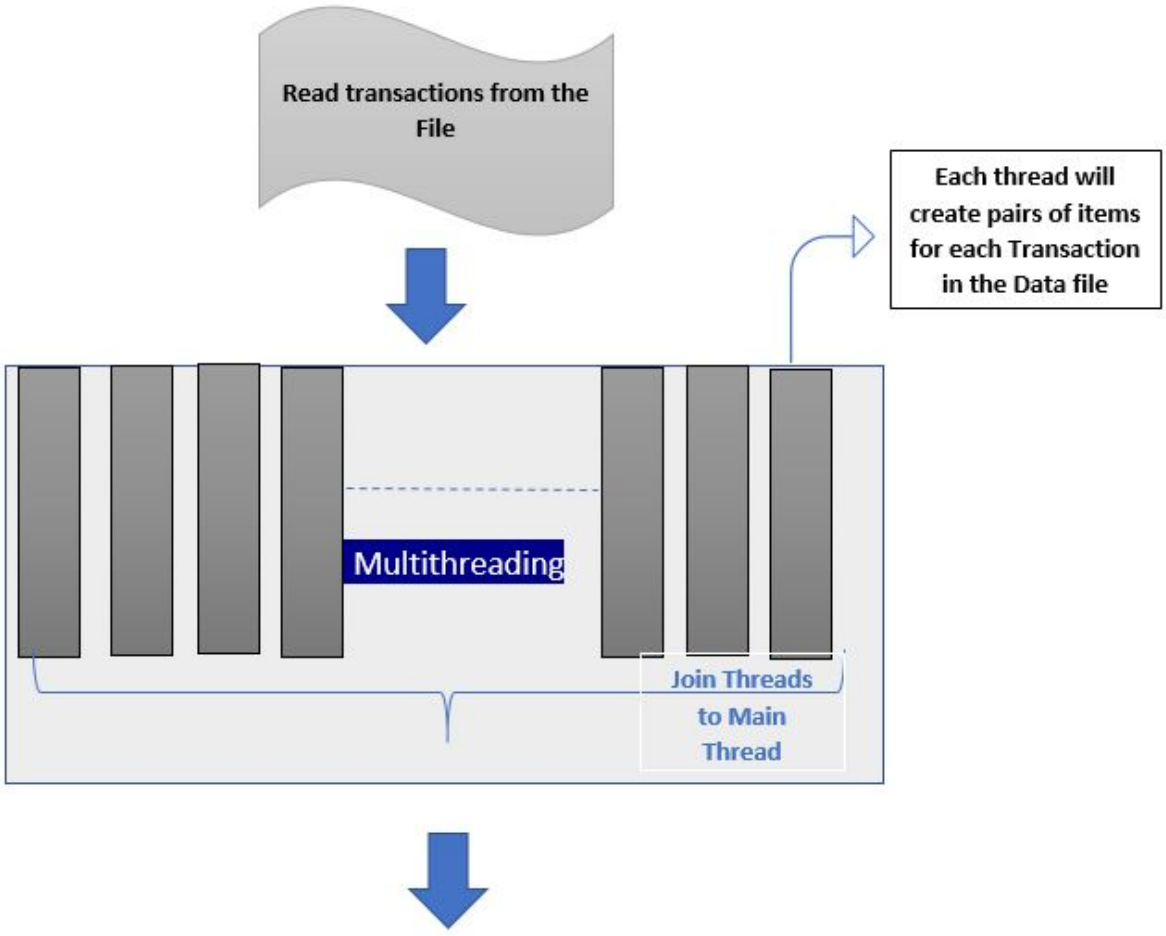
1. init : The init function initializes the thread with specified attributes.
2. run : The run function specifies what function should be executed when thread is executed.
3. To avoid two thread to overwrite a variable, threadLock object is used. To keep a variable safe from overwriting thread lock is acquired and released before and after modifying the variable .

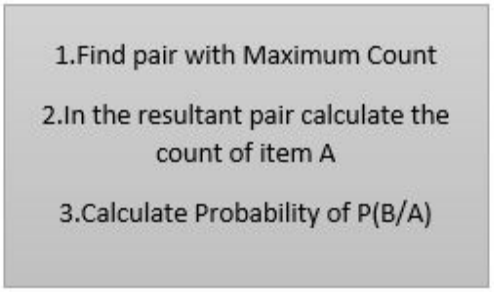
```
threadLock.acquire()
//Modify the variable
threadLock.release()
```

Steps Involved :

1. Open the file to read the transactions save these transactions in a python list
2. Each transaction is given to a thread to form pairs
3. When all the pairs are created, they are inserted into a Hash table like data structure called Dictionary. Whenever new pair is inserted the pair is inserted with count value 1 , on existing pair the count value is increased by 1.
4. Pair with Highest count is selected.Pair is considered to be Pair(A,B)
5. To calculate the conditional Probability P(A/B)= Count (A,B)/Count(A)
6. For triples it is P(A/B,C) = Count(A,B,C)/Count(B,C) is calculated and reported.

Framework of my work :





* I have tried dividing the dataset into 10 equal parts created 10 threads to do the job of pairing the transactions, but the above approach was more efficient than that.

Results:

Use Case 1: Output

```
/usr/bin/python2.7 "/home/jaya/PycharmProjects/Assignment2 OS/Trans_2_Pairs.py"
**** Phases of Execution ****
1.Opening File
('2.Number of lines in the Transaction Data file : ', 88162)
3.Closing File
4.Started Created Threads
4.Started Closing Threads
5.Completed Joining threads
6.Pairs Dictionary is formed
-------------- The Report-------------------------
('The maximum is :', (39, 48), 29142)
Probability of P(48/39) ->
[29142/50675] ->
(' Probability is -> ', 0.5750764676862358)
('Number of CPU Cores Count:', 1)
('Number of threads created :', 88161)
('Execution time: ', 23.579328060150146)
-----------------------------------------------
Exiting Main Thread

Process finished with exit code 0
```

Use Case 2:

```
****  Phases of Execution ****
1.Open file to read the Transactions
Number of lines in the Data file : 88162
2.Close the file
3.Started Creating threads
4.Started Joining threads
5. Completed Joining threads
6. Pairs Dictionary is Created
----------- The Report -------------
The Maximum occurrence triplet is : (39, 41, 48)
Frequency count : 7366
Count(A,B,C)/Count(B,C) 7366 88158
Probability of P(A,B,C) : 0.08355452709907212
Number of Cores of CPU: 4
Execution Time : 98.69027137756348
Exiting the Main thread Execution
```

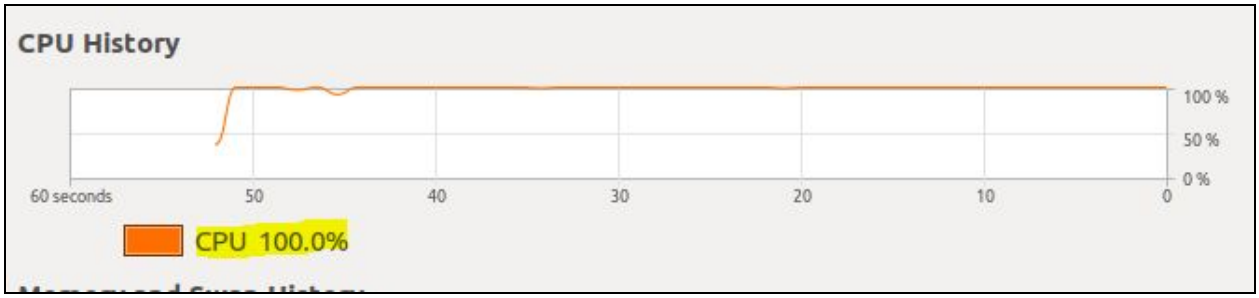
Concepts Learnt : Multithreading

1.Multithreading is a powerful mechanism which attains concurrency and parallelism.
 2.It utilises the machine resources to the maximum extent to complete the task many times faster.
3. Threads are lightweight and share same memory space as process so whenever we make changes in thread we need to make sure that it is properly modify without overwriting the values of other processes or threads. In order to do this locking is implemented.
4. Before exiting the main thread we should make sure that all the child threads created during the execution should be exited and joined back into the main thread , if not they become

Extra Credit Task:

Core 1

All threads are assigned to single CPU Core which is running at 100 % of its capacity.

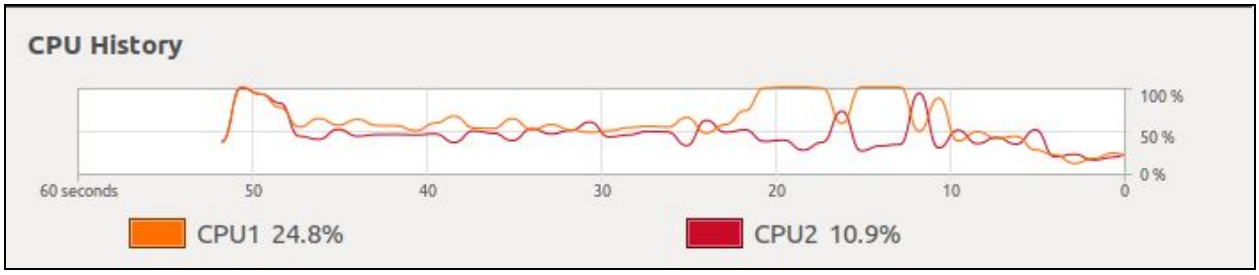


```
/usr/bin/python2.7 "/home/jaya/PycharmProjects/Assignment2 OS/Trans_2_Pairs.py"
('Count of lines', 88162)
Completed Joining threads
Pairs Dictionary is formed
('The maximum is :', (39, 48), 29142)
------------- Report------------------------
Probability of P(48/39) ->
[29142/50675] ->
(' -> ', 0.5750764676862358)
('No of CPU Cores Count:', 1)
('No of threads created :', 88161)
('The execution time in One core CPU: ', 53.354153871536255)
-------------------------------------------
Exiting Main Thread

Process finished with exit code 0
```

Core 2:

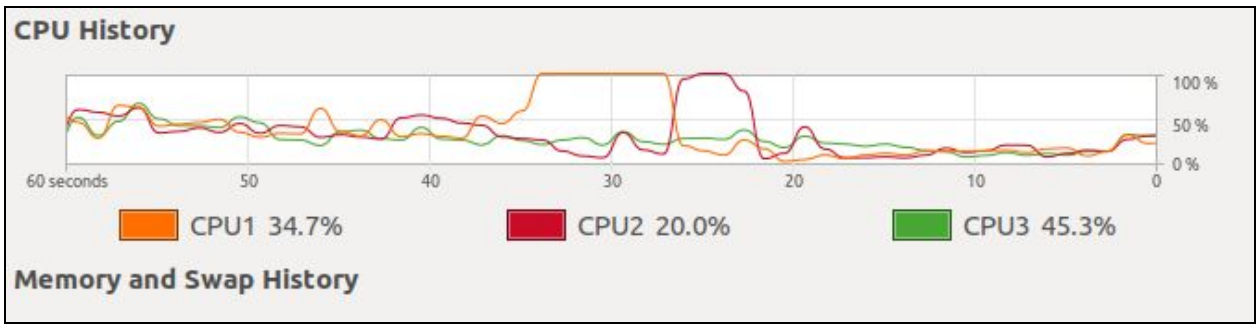
Here two CPU cores are assigned to the machine, the first half of the execution has multithreading and half of the threads are assigned to CPU 1 and half to CPU2 , so both run at an 50 % of the capacity. In the second half of the execution all the threads are joined to the main thread , there is a single thread executing the function of adding the pairs to a Dictionary table, as it is a single process thread the Orange thread (CPU1) is running at 100% of its capacity as it cannot divide the work to second CPU as thread is not divide to execute in multi threads . When the execution is complete both the cores run at below 50% capacity.



```
/usr/bin/python2.7 "/home/jaya/PycharmProjects/Assignment2 OS/Trans_2_Pairs.py"
('Count of lines', 88162)
Completed Joining threads
Pairs Dictionary is formed
('The maximum is :', (39, 48), 29142)
-------------- Report------------------------
Probability of P(48/39) ->
[29142/50675] ->
(' -> ', 0.5750764676862358)
('No of CPU Cores Count:', 2)
('No of threads created :', 88161)
('The execution time in One core CPU: ', 42.70793914794922)
------------------------------------------------
Exiting Main Thread

Process finished with exit code 0
```

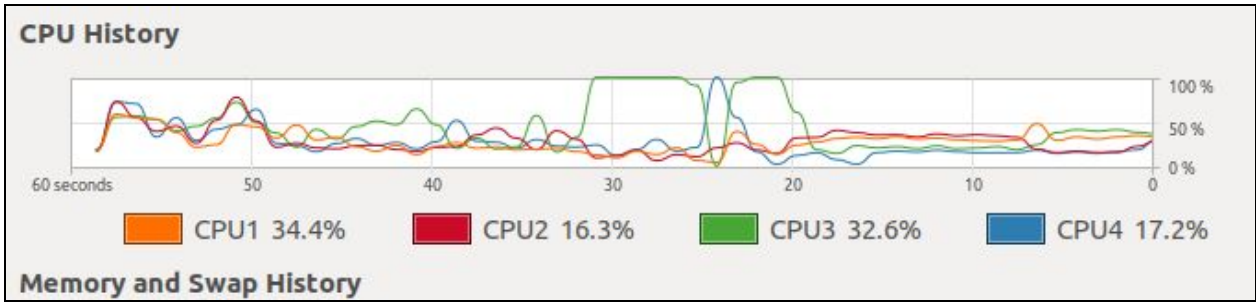
Core 3:




```
/usr/bin/python2.7 "/home/jaya/PycharmProjects/Assignment2 OS/Trans_2_Pairs.py"
('Count of lines', 88162)
Completed Joining threads
Pairs Dictionary is formed
('The maximum is :', (39, 48), 29142)
-------------- Report------------------------
Probability of P(48/39) ->
[29142/50675] ->
(' -> ', 0.5750764676862358)
('No of CPU Cores Count:', 3)
('No of threads created :', 88161)
('The execution time in One core CPU: ', 29.719595909118652)
------------------------------------------------
Exiting Main Thread

Process finished with exit code 0
```

Core 4 :




```
/usr/bin/python2.7 "/home/jaya/PycharmProjects/Assignment2 OS/Trans_2_Pairs.py"
('Count of lines', 88162)
Completed Joining threads
Pairs Dictionary is formed
('The maximum is :', (39, 48), 29142)
-------------- Report------------------------
Probability of P(48/39) ->
[29142/50675] ->
(' -> ', 0.5750764676862358)
('No of CPU Cores Count:', 4)
('No of threads created :', 88161)
('The execution time in One core CPU: ', 27.902577877044678)
-----------------------------------------------
Exiting Main Thread

Process finished with exit code 0
```

Hence time is inversely proportional to CPU cores. As CPU cores increases execution time decreases for multithreaded application.

Execution of : CPU1 > CPU2 > CPU3> CPU4