Code Details

NAME : LAKSHMI MAHURI YARAVA
ROLL : 2018101116

I have implemented Basketst Queue in two ways.

1)basketsqueue.cpp
2)cds.cpp

In first method , i tried to implement in CPP as there are no pointers in Java , so i tried my best in implementing it ,but couldn't succesfully implement it.I have used the same method as discussed in Report

Second Method:-

1. Running of code :

1. Give the number of nodes as input

2. for each node give the number of elements followed by all the elements in that node that are to be pushed

3. give the number of queries for deque

4. for each query give the number of elements to be dequed

5. give the elements to be popped followed by the index of node at which it is present

6. the element must be present in the queue at that particular time else error is thrown and exited

7. all the indices to be given are 1 based

2. Explanationd and Working :

1. when all the elements in the node are given all the elements then pushing starts

2. Here threading is intended to use , where it takes a random element from the ith node and combinely pushes all that elements into the main queue which is implemented as a vector of queues

3. So all the elements are put into queue and the elements are taken at random form the ith node if that element is still not pushed to keep track of it we remove the same from the heap produced at the begining

4. so once the heap becomes empty the program stops and deque operation comes and here multiple elements for popping are taken and similar to above threading is intended to use to make it concurrency .

5. so once after the programe has taken all the queries it exits every time the new queue is printed so as to put it in track of

3. Data Structures Used :

1. for heap : an array of vectors to keep track of it

2. for taking random numbers : rand(time(0))

3. main queue : vector of vectors to make it simpler for popping at certain indices too else the complexity will raise a lot

4. for storing the queries : pair of vectors

4. Complexity :

1. for enque : O(n) where n is number of total elements

2. for deque : O(logn) in average O(n) at worst case

5. Main theme and langauge :

1. Threading is intended to use but couldnot do that as cpp is very difficult for threading processes and the java is not accessible for the pointers as mentioned in the paper so i have used cpp for this

2. Here delays are used for obseving the outputs and if th threading is to be used it should placed just above at all the places where the delays are used

6. Conclusion and optimisation:

1. The main optimisation can be done using CAS or THREADING which are very fast in concurrency

2. Because of the pointers issue in java and Threading issues in cpp brute force is coded

Paper Link :

https://people.csail.mit.edu/shanir/publications/Baskets%20Queue.pdf