Assignment 1

Venkata Tejaswini Anagani - EE18BTECH11047

Download all the codes and figures from

1 PROBLEM

A queue is implemented using a non-circular singly linked list. The queue has a head pointer and a tail pointer, as shown in the figure. Let n denote the number of nodes in the queue. Let *enqueue* be implemented by inserting a new node at the head, and *dequeue* be implemented by deletion of a node from the tail.
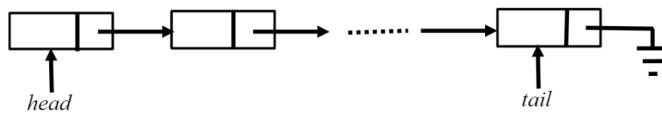

head   *tail*

Fig. 0

What is the time complexity of the most time-efficient implementation of *enqueue* and *dequeue*, respectively, for this data structure?

2 SOLUTION

Enqueue: The enqueue operation involves two operations:

- Create a new node with the value to be inserted and point the next pointer of this node to the head of the linked list.
- Make the new node as the head of the linked list.

Creating a new node and changing two pointers can be performed in constant time. Therefore, time complexity of *enqueue* operation is $\Theta(1)$.

Dequeue: The dequeue operation involves the following steps:

- Delete the tail node.
- Traverse the linked list to find the address of the second last node and make the next pointer NULL. Update the tail pointer.

Since we are traversing the whole linked list, the time complexity of *dequeue* operation is $\Theta(n)$ where n is the number of nodes in the linked list.

Verification:
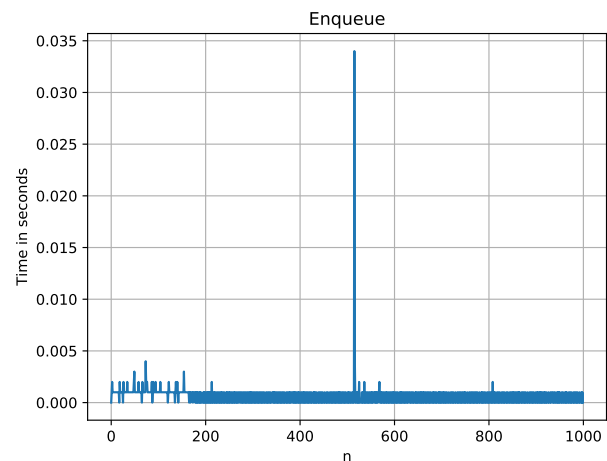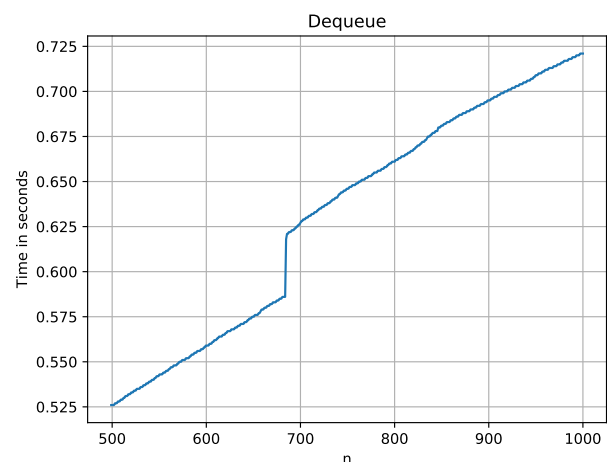Below are the experimental plots of time complexities.

Fig. 0: 1

Fig. 0: 2

The following code executes enqueue and dequeue

operations several times and stores the execution time values in a dat file.

https://github.com/anaganitejaswini/C−and−DS/
gate_assignment/codes/linked_list.c

The following python code plots the time complexity of enqueue and dequeue.

https://github.com/anaganitejaswini/C−and−DS/
gate_assignment/codes/time_complexity.py

Therefore, the time complexity of the most time efficient implementation of *enqueue* and *dequeue* respectively are $\Theta(1)$ and $\Theta(n)$.