MA 251 Data Structures

Laboratory Assignment 6

25-09-2019

Note: Upload your programs to the server (deadline: 4:30 pm)

1. **Merge sort with arrays**: Write a program to implement merge sort using arrays. As discussed in class, the procedure MERGE-SORT(A, p, r) sorts the elements in the subarray A[p .. r]. The pseudocode of Merge sort is given below.

```
MERGE-SORT(A, p, r)
if p < r
    q = ⌊(p + r)/2⌋
    MERGE-SORT(A, p, q)
    MERGE-SORT(A, q + 1, r)
    MERGE(A, p, q, r)
```

The procedure of MERGE() is given overleaf. Your code should be implementation of these pseudocodes.

In the main function,

- populate an array **A** with **n** *random* integers in the range [0, 10^3-1]. Take **n=10^{k}**, where k = 1, 2 and 3.
- Call Merge sort for the three different values of k and compute the time taken by each call.
- For each value of k, print the unsorted array (row wise in line 1), sorted array (row wise in line 2) and the time taken (in third line).

Hint:
The *random* nos. can be generated by the *rand*() function. Do not forget to *seed* the *rand*() function with the current time.

The predefined *clock*() function can be used to measure the running time. An example code segment is

```
c1 = clock();
…. Call Merge sort …
c2 = clock();
runtime = c2 – c1;
```

For small values of **n**, it is likely that you will get the runtime to be 0. In order to ensure a more accurate timing measurement, for a particular value of **n** (say 10), perform the sort experiment 1000 times in a row and then divide the total elapsed time by 1000. This strategy gives you a timing measurement that is much more accurate.

2. **Merge sort with linked list**: Repeat the above assignment using linked lists.

Pseudocode of MERGE

```
MERGE(A, p, q, r)
n1 = q − p + 1
n2 = r − q
let L[1 .. n1 + 1] and R[1 .. n2 + 1] be new arrays
for i = 1 to n1
        L[i] = A[p + i − 1]
for j = 1 to n2
        R[j] = A[q + j]
L[n1 + 1] = ∞
R[n2 + 1] = ∞
i = 1
j = 1
for k = p to r
        if L[i] ≤ R[j]
                A[k] = L[i]
                i = i + 1
        else A[k] = R[j]
                j = j + 1
```