Data Management Assignment 5

Cuihuan Zhang & Janis Waser

May 11, 2025

A. 1) (a) If we are inserting at the beginning of a heap it takes only 1 operation.

(b) If we have no pointer to the back, it takes N operations to get there and insert the value.

2) It takes log N operations to find the position where the value has to be inserted if the sequence is balanced (no overflow values). Unless the inserted value belonged to the end for example if a new maximum was inserted in which case the moving would not be necessary, every value which position is after in the sequence (tail) has to be moved taking N operations. Added up this gives us log N + N operations.

3) In a 2-3 tree it always takes O(log N) operations to insert a value. Either the inserted value can be positioned at the lowest level for which one has to go down the tree to find the place, and since the tree is guaranteed to be balanced it takes exactly log N steps.

The other case is where there is no place at the lowest level and therefore value have to propagate up, but again it takes log N operations in the worst case to go up the entire tree this gives us log N + log N steps which in the big-O notation is also O(log N).

B. 1) If for every key there exists a pointer to the file where the key is stored, we call it a dense index, sparse otherwise. So it is a dense index if every key is indexed.

2) If every key that is in between two indexed keys is stored in the same file then we call it a clustered file, if this is not the case then it is unclustered.

C.

D. The first option is to read R once and compare with S which needs to be done 1000 times (2000/2), this gives read R + read 500/4 S= 2000/2 + 1000*4000/4=1'001'000.

The other option is where you read S first and compare 1000 (4000/4) times with R this gives read S+ read 1000 R= 4000/4+ 1000*2000/2=1’001’000. The second option is considerably worse hence we opt to read R and compare with S.