

Ph.D. Qualifying Exam (Fall 2000)

Algorithms

Answer ALL questions

Question 1. (30%)

Consider a rectangular array. Sort the elements in each row into increasing order. Next sort the elements in each column into increasing order. Argue that the elements in each row remain sorted.

Question 2. (35%)

Let $T(n) = T(an) + T(bn) + T(cn) + \Theta(n)$, where $0 < a$, $0 < b$, $0 < c$ and $a+b+c < 1$. Show that $T(n) = \Theta(n)$. That is, you have to show that $T(n) = O(n)$ and $T(n) = \Omega(n)$.

Question 3. (35%)

Design a data structure that supports the following set operations on integers.

1. `NewSet()`,
which returns a pointer to an empty set of integers.
2. `Insert(S, x)`,
which inserts into the set S a new element of the integer value x .
Example: `Insert({ 5, 9, 30 } , 9) = { 5, 9, 9, 30 }`.
3. `Delete(S, x)`,
which deletes from S an element of the integer value x .
If x does not exist, then S is not changed.
Examples:
`Delete({ 5, 9, 9, 30 } , 5) = { 9, 9, 30 }`.
`Delete({ 5, 9, 9, 30 } , 9) = { 5, 9, 30 }`.
`Delete({ 5, 9, 9, 30 } , 7) = { 5, 9, 9, 30 }`.
4. `Print_k_Smallest(S, k)`,
which prints in ascending order the k smallest elements in S .
Examples:
`Print_k_Smallest({ 9, 20, 7, 4, 9, 3, 8, 12 } , 5)` prints 3, 4, 7, 8, 9.
`Print_k_Smallest({ 9, 20, 7, 4, 9, 3, 8, 12 } , 7)` prints 3, 4, 7, 8, 9, 9, 12.
`Print_k_Smallest({ 9, 20, 7 } , 5)` prints 7, 9, 20.

The running time for `NewSet()`, `Insert(S, x)`, `Delete(S, x)` and `Print_k_Smallest(S, k)` should be $O(1)$, $O(\lg \#S)$, $O(\lg \#S)$ and $O(k)$ respectively.

Sketch of answers:

Question 1.

Let $A(i, j)$ be the elements of the rectangular array after the elements are sorted in each row into increasing order. Next, we sort the elements in each column. Suppose the element $A(i, j)$ is ranked k in the j -th column during the column sorting phase. Let $A(i_1, j), A(i_2, j), \dots, A(i_{k-1}, j)$ be the $k - 1$ elements in the j -th column that are ranked higher than $A(i, j)$. Then $A(i_1, j - 1), A(i_2, j - 1), \dots, A(i_{k-1}, j - 1)$ and $A(i, j - 1)$ are k elements in the $(j - 1)$ -th column that are of values at most $A(i, j)$. That is, the k -th ranked element in the $(j - 1)$ -th column is not larger than the corresponding k -th ranked element $A(i, j)$ in the j -th column.

Question 2.

The proof is similar to that of Select, which proves that $T(n) = O(n)$ where $T(n) = T(n/5) + T(7n/10) + O(n)$. In addition, one has to show that $T(n) = \Omega(n)$. This is immediate since by definition $T(n) = T(an) + T(bn) + T(cn) + \Theta(n)$ which is $\Omega(n)$.

Question 3.

Use a red-black tree. In addition, maintain a linear doubly-linked list of all the elements in ascending order. Thus, an element in a set is represented in both structures. The same element in both structures are linked by pointers.

Implementation details for $\text{Insert}(S, x)$:

After inserting x into the red-black tree, we search for the predecessor y of x in S in the red-black tree which takes logarithmic time. Next, follow the link from the element y to the same element in the linear doubly-linked list. Then insert a new element for x into the linear list. Finally, set up the link between the element x in the red-black tree and the element x in the linear list. (If the predecessor does not exist, then x is the new smallest element. The treatment is easy.)