
T1:
\[ T(n) \leq 3 T\left(\frac{n}{2}\right) + n^2 \]

By master theorem, \( T(n) = O(n^2) \).

2. \( T(n) \leq 5 T\left(\frac{n}{2}\right) + n^2 \)
   
   By MT: \( T(n) = O\left(n^{\log_2{5}}\right) \)

3. \( T(n) \leq 2 T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) \)

Let \( T(n) = cn^k \), \( k \) to be chosen to satisfy the inductive step & \( c \) to be chosen to satisfy the base case.

To satisfy the inductive step
need: \( 2c\left(\frac{n}{2}\right)^k + c\left(\frac{n}{2}\right)^k \leq cn^k \)

i.e. need: \( 2\left(\frac{1}{4}\right)^k + \left(\frac{1}{2}\right)^k \leq 1 \)

\( k \approx 0.86 \)

Hence \( T(n) = O(n^{0.86}) \)

Note: I meant to ask for
\[ T(n) \leq 2 T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n \]

Then since \( 2\left(\frac{1}{4}\right)^k + \left(\frac{1}{2}\right)^k < 1 \), \( T(n) = O(n) \).

This can be proved by induction.

Some students proved this bound for the given recurrence by induction: \( T(n) \leq cn \) is not a tight bound for the given recurrence, but we have given credit for the solution.
4. \( T(n) \leq 2n T(n-1) \)
\[ \leq 2n \cdot 2(n-1) T(n-2) = 2n \cdot 2 \cdot n(n-1) T(n-3) \]
\[ \leq 2^{n-1} n(n-1) \cdots 3 \cdot 2 \cdot T(1) \]
Hence \( T(n) = O(2^n n!) \).

5. \( T(n) \leq 2T(n-1) + T(n-2) \)
Assuming it to be a quadratic and \( T(n) = c^n \)
\[ c^n = 2c^{n-1} + c^{n-2} \]
\[ i.e. \quad c^2 - 2c - 1 = 0 \]
\[ i.e. \quad c = 1 \pm \sqrt{2} \]
Hence \( T(n) = a (1+\sqrt{2})^n + b (1-\sqrt{2})^n \)
since \( (1-\sqrt{2})^n = O(1) \).

III
\[ c[i,j] \quad x_i \ldots x_j \]
\[ y_i \ldots y_j \]
If \( x_i \) couples with \( y_j \) then \( c[i-1,j-1] + s(x_i,y_j) \).
If \( x_i \) couples with a symbol \( y_k \) \( i < j \) then \( c[i,j-1] \).
Hence \( c[i,j] = \max \left\{ c[i-1,j-1] + s(x_i,y_j), c[i,j-1] \right\} \) if \( i < j \).
\[ c[1,1] = \max \left\{ s(x_1,y_1), c[1,j-1] \right\} \]
Base: \( c[1,j] = s(x_1,y_j) \)
Compute \( c[1,j], c[2,j], \ldots, c[n,j], c[1,2], \ldots \); i.e.
in increasing value of \( j-i \).
Each \( c[i,j] \) requires \( O(1) \) steps.
Hence speed \( = O(n(m-n)) = O(nm) \).
4(a) Start at the root with REN = k.

Repeat
  At any node, access the size of the left subtree
  If size ≥ k move to the left child
  If size = k - 1, return the value at the root
  If size < k
    \[ \text{REN} = \text{REN} - \text{size} - 1 \]
    move to the right child

1. b) First compute the rank of x,
    Then compute the rank of y,
    Then return \[ \text{rank}(y) - \text{rank}(x) + 1 \].

Computation of the rank of a given number \( z \):
start at the root, set count = 0;
follow the path for \( z \) as in any binary search tree;
if right child is taken, count = count + size(left child) + 1;
if it stops at the node, count = count + 1 2 (if not count

IV 2) 6)

a) False since a balanced binary tree is never more space efficient.

b) False since if a BBT can never require less space, it usually takes more space since you need to store some flags. For example, in a Red-Black tree you need to store a bit for the color. Also, a complete BBT can be stored in a RAM as an array without losing pointers.