בס"ד

Leonard Blam
900086

Homework 3 Theoretical HW DS 2
Answers are in **bold**

Ex3: Recurrences

1. Given the following recursive function

    f(*V*, *n*) // V is an array, $n \geq 1$
    if *n*< 2
    return 0;
    else
    { $m = \left\lfloor \frac{n}{2} \right\rfloor$;
    return *(f (V,m)+f(V,m));*
    }

    a. Write a recurrence describing the function's runtime.

    (In what I understand the question is by writing out the runtime as T(n))

    $$\boldsymbol{T(n) = 2T\left(\frac{n}{2}\right) + 1}$$

    b. Solve the recurrence, and write the runtime as a function of n.

    $$\boldsymbol{a = 2, b = 2, f(n) = 1, logb(a) = 1, n^{logb(a)} = n}$$

    $$\boldsymbol{f(n) = O(n^{1-e}) => T(n) = \Theta(n)}$$

2. Given the following recursive function

    f(*V*,*from*,*to*) // V is an array, $from, to$ are indices
    if (*from>=to)*
    return true;
    else if (V[from] !=V[to])
    return false;
    else return f(V,from+1,to) and $g(V, from, to)$;

    a. Assume $g(V, from, to)$ scans V's elements with indices in $[from, to]$ once.

    Write a recurrence describing the function's runtime.

    (In what I understand the question is by writing out the runtime as T(n))

    T(n) = T(n-1) + n

    b. Solve the recurrence, and write the runtime as a function of n.

    T(n) = O(n^2)

3. Solve the following recurrences, assuming $T(1) = 1$.

a) $T(n) = 2T(n/2) + n^3$

a = b = 2, f(n) = n^3, logb(a) = 1, n^logb(a) = n
f(n) = Ω(n^1+e) => T(n) = Θ(n^3)

b) $T(n) = T(n/3) + 2\log n$

a = 1, b = 3, f(n) = 2log(n), logb(a) = 0, n^logb(a) = 1
f(n) = Ω(n^e) => T(n) = Θ(2log(n)) = Θ(log(n))

c) $T(n) = T(n/8) + nlogn$

a = 1, b = 3, f(n) = nlog(n), logb(a) = 0, n^logb(a) = 1
f(n) = Ω(n^e) => T(n) = Θ(nlog(n))

d) $T(n) = 2T(n/3) + n^{1.5}$

a = 2, b = 3, f(n) = n^3/2, logb(a) = 0.6, n^logb(a) = n^0.6
f(n) = O(n^0.6-e) => T(n) = Θ(n^3/2)

e) $T(n) = 3T(n/2) + n^2logn$

a = 3, b = 2, f(n) = n^2 log(n), logb(a) = 3/2, n^logb(a) = n^3/2
f(n) = Ω(n^1.5 +e) => T(n) = Θ(n^2 log(n))

f) $T(n) = 2T(n/4) + \sqrt{n}$

a = 2, b= 4, f(n) = root(n), logb(a) = ½, n^logb(a) = root(n)
f(n) = Θ(root(n)) => T(n) = Θ(root(n)log(n))

g) $T(n) = T(\sqrt{n}) + 2logn$

T(n) = O(logn)

h) $T(n) = 4T(n/4) + n / \lg n$

a = 4, b = 4, f(n) = n/log(n), logb(a) = 1, n^logb(a) = n
f(n) = O(n^1-e) => T(n) = Θ(n)