Homework #3

CS 4120-5120 Due: Mon, Oct 25, 2021

Fall, 2021 30 points

Unless clearly indicated otherwise, everybody should answer every part of every question.

1. ~~Describe~~ how you could create an efficient algorithm *based on quicksort* for the following problem. A simple way to accomplish this task would be to sort the array completely, but your algorithm can be more efficient if you do not attempt to completely sort the list! Given a list of *n* distinct positive integers, rearrange the list into two sublists, each of size *n*/2, such that the difference between the sums of the integers in the two sublists is maximized. You may assume that *n* is a multiple of 2 (i.e. *n* is even). Although you need not give a precise algorithm (unless you like to do so), determine its time complexity based on the master theorem.

One way is to sort the entire list of *n* distinct integers. Split the list cleanly in half. This way the left list will all be numbers less than the right side of the list, that will maximize the difference calculated from each side because all integers smaller than the list/2 is on one side and all the larger number are on the other side. The Time complexity will be that of O(nlogn) as a worst case and O(n) in the best case.

1. ~~Use~~ Strassen’s algorithm to compute the matrix product (no credit if you do not show ALL steps or if it becomes clear you have not used Strassen’s algorithm):

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | 1 | 3 | 4 |
| 4 | 2 | 7 | 5 |

|  |  |  |  |
| --- | --- | --- | --- |
| A11 | A12 | B11 | B12 |
| A21 | A22 | B21 | B22 |

S1 = -1

S2 = 3

S3 = 6

S4 = 4

S5 = 4

S6 = 8

S7 = -1

S8 = 12

S9 = -2

S10 = 7

P1 = -2

P2 = 15

P3 = 18

P4 = 8

P5 = 32

P6 = -12

P7 = -14

C11=32+8−15+-12=13

C12=-2+15=13

C21=18+8=26

C22=32+-2−18−-14=26

1. ~~As~~ described, Strassen’s algorithm works for exact powers of 2. We could modify Strassen’s algorithm to multiply two *n*  *n* matrices, where *n* is not an exact power of 2 as follows: We add as many zeroes at the end of each row and column as needed to turn the matrix into a larger square matrix, whose number of rows (and columns) would be the next higher power of 2, and extract the first n rows and columns from the result! Argue that the resulting algorithm also runs in Θ(*n*lg 7) time.

Strassen’s algorithm recursion is T(n) = 7T(n/2) + Θ(n2)

Each extended matrix will show 3x3 = T(n) = kT(n/3) + Θ(n2)

4x4 = T(n) = kT(n/4) + Θ(n2)

Etc…

To know the complexity, you just need to extract the matrix from C’ then you can determine if the resulting algorithm also runs at the same Θ time.

So, for this method it is Θ(m^lg7) because 2^(k-1) < n and m < 2n

Θ(2n^(lg7)) = Θ(2^(lg7) \* n^(lg7)) = Θ(n^(lg7))

1. ~~Show~~ how the *Heapsort* algorithm works on the following numbers, by showing the binary tree view after each step of Build-Max-Heap, and for the rest of Heapsort. [Hint: Do not get sloppy as you approach the end; show your work completely.]

60 30 50 80 10 70 40 90 20 100

A picture containing application

Description automatically generatedShape

Description automatically generatedShape

Description automatically generatedShape

Description automatically generatedShape

Description automatically generated with low confidenceDiagram

Description automatically generatedA picture containing shape

Description automatically generatedA picture containing diagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generated with medium confidenceDiagram, engineering drawing

Description automatically generatedDiagram, engineering drawing

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generated with medium confidenceDiagram

Description automatically generated

1. ~~Exercise~~ 12.1-1, page 289 of your text.

For the set of keys {1, 4, 5, 10, 16, 17, 21}, draw binary search trees of height 2, 3, 4, 5, and 6.

A picture containing hanger

Description automatically generated

Diagram

Description automatically generatedDiagram

Description automatically generated

Height 2

Height 3 Height 4

Diagram

Description automatically generatedDiagram

Description automatically generated

Height 5 Height 6

1. ~~a~~) [**CS 4120 ONLY**] Is 5n little-o, θ, or little-ω of 7n? Justify your answer mathematically. NOW, take the natural (not binary) logarithms on both sides, and answer whether ln 5n is little-o, θ, or little-ω of ln 7n. Once again, justify your answer mathematically. (You may still check out b for learning better.)

b) [**CS 5120 ONLY**~~] Suppose c and k are positive constants such that 1 < c < k. Is c~~~~n~~ ~~little-o, θ, or little-ω of k~~~~n~~~~? Justify your answer mathematically. NOW, take the natural logarithms on both sides, and answer whether ln c~~~~n~~ ~~is little-o, θ, or little-ω of ln k~~~~n~~~~. Once again, justify your answer mathematically.~~

Little-o if lim f(n)/g(n) = 0

If f(n) is theta of g(n), then the value f(n) is between c1\*g(n) and c2\*g(n)

Little- ω if lim f(n)/g(n) = ∞

f(n) = 5^n g(n) = 7^n

f(n)<g(n) where n0 = 1 and c =1

∴ f(n) = o(7^n)

F(n) = ln(5^n) == n ln 5

G(n) = ln(7^n) == n ln 7

~~n~~ ln 5 <= ~~n~~ ln 7 == ln 5 <= ln 7

t(n) <= c2 g(n) where c2 = 1

~~n~~ ln 7 <= 2 ~~n~~ ln 5 == ln 7 <= 2ln5

== ½ ln 7 <= ln5

c1 = ½ , c2 = 1

== θ (ln 7^n)