×

Problem Set #1

Back to Week weekNumber

4/5 points earned (80%)

Not Verified You didn't verify this quiz. Retake with verification to be eligible for a Course Certificate.

1 / 1
points

1.
3-way-Merge Sort : Suppose that instead of dividing in half at each step of Merge Sort, you divide into thirds, sort each third, and finally combine all of them using a three-way merge subroutine. What is the overall asymptotic running time of this algorithm? (Hint: Note that the merge step can still be implemented in $O(n)$ time.)

- n
- $n \log(n)$

Correct Response
That's correct! There is still a logarithmic number of levels, and the overall amount of work at each level is still linear.

- $n(\log(n))^2$
- $n^2 \log(n)$

---

1 / 1
points

2.
You are given functions f and g such that $f(n) = O(g(n))$. Is $f(n) * log_2(f(n)^c) = O(g(n) * log_2(g(n)))$? (Here c is some positive constant.) You should assume that f and g are nondecreasing and always bigger than 1.

- Sometimes yes, sometimes no, depending on the constant c
- True

Correct Response
That's correct! Roughly, because the constant c in the exponent is inside a logarithm, it becomes part of the leading constant and gets suppressed by the big-Oh notation.

- Sometimes yes, sometimes no, depending on the functions f and g
- False

---

1 / 1
points

3.
Assume again two (positive) nondecreasing functions f and g such that $f(n) = O(g(n))$. Is $2^{f(n)} = O(2^{g(n)})$? (Multiple answers may be correct, you should check all of those that apply.)

- [ ] Sometimes yes, sometimes no (depending on f and g)

Correct Response

- [ ] Always

Always

Correct Response

What if f(n) = 2n and g(n) = n?

☐ Never

Correct Response

For example, what if f(n)=g(n)?

☐ Yes if $f(n) \leq g(n)$ for all sufficiently large n

Correct Response

---

1 / 1 points

4.

k-way-Merge Sort. Suppose you are given k sorted arrays, each with n elements, and you want to combine them into a single array of kn elements. Consider the following approach. Using the merge subroutine taught in lecture, you merge the first 2 arrays, then merge the 3^{rd} given array with this merged version of the first two arrays, then merge the 4^{th} given array with the merged version of the first three arrays, and so on until you merge in the final (k^{th}) input array. What is the running time taken by this successive merging algorithm, as a function of k and n? (Optional: can you think of a faster way to do the k-way merge procedure ?)

○ $\theta(nk^2)$

Correct Response

That's correct! For the upper bound, the merged list size is always $O(kn)$, merging is linear in the size of the larger array, and there are k iterations. For the lower bound, each of the last $k/2$ merges takes $\Omega(kn)$ time.

○ $\theta(n\log(k))$

○ $\theta(nk)$

○ $\theta(n^2k)$

✖ 0 / 1 points

5.
Arrange the following functions in increasing order of growth rate (with $g(n)$ following $f(n)$ in your list if and only if $f(n) = O(g(n))$).

a)$\sqrt{n}$

b)10^n

c)$n^{1.5}$

d)$2^{\sqrt{\log(n)}}$

e)$n^{5/3}$

Write your 5-letter answer, i.e., the sequence in lower case letters in the space provided. For example, if you feel that the answer is a->b->c->d->e (from smallest to largest), then type abcde in the space provided without any spaces before / after / in between the string.

You can assume that all logarithms are base 2 (though it actually doesn't matter).

WARNING: this question has multiple versions, you might see different ones on different attempts!

Preview

$adceb$

Please note: Each of the following will be interpreted as a single variable, not as a product of variables: adceb. To multiply variables, please use * (e.g. enter x*y to multiply variables x and y).

adceb

Incorrect Response

One approach is to graph these functions for large values of n. Once

One approach is to graph these functions for large values of n. Once in a while this can be misleading, however. Another useful trick is to take logarithms and see what happens (though again be careful, as in Question 3).

Correct Answer: daceb

One approach is to graph these functions for large values of n. Once in a while this can be misleading, however. Another useful trick is to take logarithms and see what happens (though again be careful, as in Question 3).