CMSC 351 (JWG) Exam 2 Spring 2021

Solutions

Exam Logistics:

1. From the moment you download this exam you have three hours to take the exam and submit to Gradescope. This includes the entire upload and tag procedure so do not wait until the last minute to do these things.
2. Tag your problems! Please! Pretty please!
3. You may print the exam, write on it, scan and upload.
4. Or you may just write on it on a tablet and upload.
5. Or you are welcome to write the answers on separate pieces of paper if other options don't appeal to you, then scan and upload.

Exam Rules:

1. You may ask for clarification on questions but you may not ask for help on questions!
2. You are permitted to use official class resources which means your own written notes, class Panopto recordings and the textbook. The textbook won't help much so don't worry if you don't have it.
3. You are not permitted to use other resources. Thus no friends, internet, calculators, Wolfram Alpha, etc. Question 3 is an exception - you can use technology there. Still no friends etc.!
4. By taking this exam you agree that if you are found in violation of these rules that the minimum penalty will be a grade of 0 on this exam.

Exam Work:

1. Show all work as appropriate for and using techniques learned in this course.
2. Any pictures, work and scribbles which are legible and relevant will be considered for partial credit.
3. Arithmetic calculations do not need to be simplified unless specified.

1. (a) Suppose the partition algorithm from QuickSort is used just once once on a list and the resulting list is: [5 pts]

$$[2, 1, 6, 9, 7, 10, 17, 16, 11, 20]$$

Which value(s) could have been the pivot? Explain.

Solution:

The possilbe pivots are 6,10,20 since these are the only elements for which everything to the left is smaller and everything to the right is larger.

(b) Draw a visual representation of MergeSort applied to the list: [5 pts]

$$[5, 6, 2, 8, 1, 7, 3]$$

See page 2 of the `mergsort.pdf` notes to see what you should be drawing here.

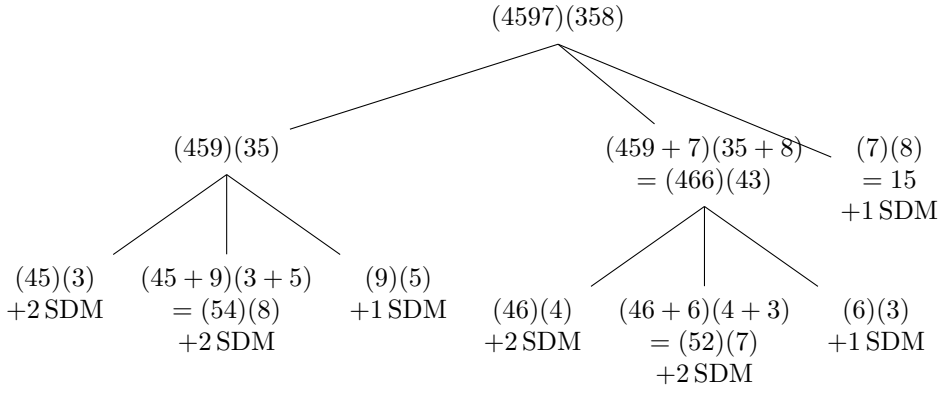
Solution:

This isn't as pretty as the notes but perhaps clear enough:

```
      [5,6,2,8,1,7,3]
   [5,6,2]       [8,1,7,3]
 [5][6,2]       [8,1][7,3]
    [6][2]   [8][1]   [7][3]
   [2,6]       [1,8][3,7]
 [2,5,6]       [1,3,7,8]
   [1,2,3,5,6,7,8]
```

2. Draw the tree diagram for Karatsuba's Algorithm applied to (4597)(358) and use it to count the number of single-digit multiplications. [10 pts]

Solution:



We have a total of 11 single-digit multiplications.

3. Suppose $T(n) = 2T(n/8) + f(n)$. Use the Master Theorem to solve this recurrence relation for each of the following $f(n)$. You do not need to check the regularity condition if that case arises.

   (a) With $f(n) = 1 + \sqrt[3]{n}$. [3 pts]

   Solution:

   We have $1 + \sqrt[3]{n} = \Theta(n^{1/3})$ and $1/3 = 1/3$. Thus Case 2 yields $T(n) = \Theta(n^{1/3} \lg n)$.

   (b) With $f(n) = n \ln n$. [3 pts]

   Solution:

   We have $n \lg n = \Omega(n^1)$ and $1 > 1/3$. Thus Case 3 yields $T(n) = \Theta(n \lg n)$.

   (c) With $f(n) = \sqrt[4]{n} + 256$. [4 pts]

   Solution:

   We have $\sqrt[4]{n} + 256 = \mathcal{O}(n^{1/4})$ and $1/4 < 1/3$. Thus Case 1 yields $T(n) = \Theta(n^{1/3})$.

4. Given the following recurrence relation:

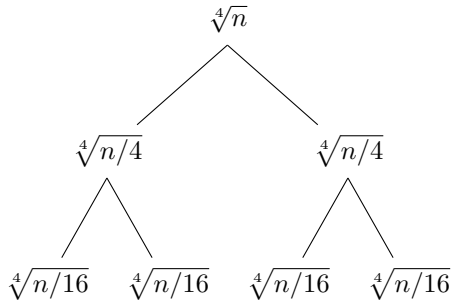
$$T(n) = 2T(n/4) + \sqrt[4]{n} \text{ with } T(1) = 3.$$

Assume for what follows that n is some power of 4. This removes any concerns about the necessity of floors and ceilings.

It's a fact that there are $1 + \log_4 n$ layers in the associated recurrence tree. If we label them $k = 0$ for the root node, $k = 1$ for the next layer, and so on, up to $k = \log_4 n$ for the leaf nodes then layer k has 2^k entries. Do not prove this.

(a) Draw the first three layers of the associated recurrence tree and fill in the expressions (involving n with all T calculated) for each node. [5 pts]

Solution:



(b) For an arbitrary non-leaf layer $0 \leq k \leq \log_4 n - 1$ what is the time requirement for each node and consequently what is time requirement for the entire layer? [5 pts]

Solution:

Each entry is $\sqrt[4]{n/4^k}$ for a total of $2^k \sqrt[4]{n/4^k}$.

(c) Following up from that, for the leaf-layer $k = \log_4 n$ what is the time requirement for each node and consequently what is the time requirement for the entire layer? [5 pts]

Solution:

The full layer would be $2^{\log_4 n}(3) = 2^{\log_2 n / \log_2 4}(3) = 3\sqrt{n}$.

(d) Write down a sum for the total time in the tree. This should absolutely involve Σ notation. You do not need to evaluate this sum. [5 pts]

Solution:

The final result would be:

$$3\sqrt{n} + \sum_{k=0}^{\log_4 n - 1} 2^k \sqrt[4]{n/4^k}$$

5. Suppose a list A of length n indexed at 1 represents a heap, write the pseudocode for an algorithm which returns a list of nodes which violate the maxheap property. [10 pts]

Solution:

We check each node to make sure that its value is greater than or equal to the value at its children. The only thing we need to be careful of is to not spill over the end of the array.

```
violators = []
for i = 1 to n
  leftchild = 2i
  rightchild = 2i+1
  if leftchild <= n and A[i] < A[leftchild]
    violators.append(i)
  elseif rightchild <= n and A[i] > A[rightchild]
    violators.append(i)
  endif
endfor
```

6. A list A contains only integer powers of 2, has minimum min and maximum max. For example $A = [2^3, 2^{-1}, 2^2]$ has $min = \frac{1}{2}$ and $max = 8$.

   (a) Explain how you could modify the CountingSort pseudocode to sort this list by sorting by exponent rather by the value itself. For example if $A = [2^3, 2^{-1}, 2^2]$ you can sort this by sorting $[3, -1, 2]$ instead. I am not expecting/wanting pseudocode, I am just asking you to give a brief overview of how you would modify the regular pseudocode. [5 pts]

   Solution:

   First take the lg of each entry and subtract $\lg(min)$ from each entry. Now the values are in the range $0, ..., \lg(max) - \lg(min)$. Do counting sort. Then do add $\lg(min)$ to each entry and take take 2 to the power of each entry.

   (b) What would the Θ time complexity be in terms of n, min and max? [5 pts]

   Solution:

   The new range is from $\lg(min)$ to $\lg(max)$ and so when we shift the values the results are from 0 to $\lg(max) - \lg(min)$ so it's $\Theta(n + \lg(max) - \lg(min))$.

   (c) Give two reasons why this is preferable to just using CountingSort directly. [5 pts]

   Solution:

   The range of values is almost certainly much smaller. We don't need to deal with fractions.

7. Suppose A is a list of n distinct positive integers and you wish to choose the kth order element. [10 pts]
You have a function `where(i,k)` which will return:

- `True` if the kth order element is at index i.
- `Left` if the kth order element is to the left of index i,
- `Right` if the kth order element is to the right of index i,

Explain how you can find the kth order element in $\lg(n)$ time. You don't need to write pseudocode, a decent explanation of the algorithm and why it's $\lg(n)$ will suffice.

Solution:

Like a midpoint rule or binary search.

Set $L = 0$ and $R = n - 1$.
Repeat the following until it's found:

- Check the middle of L and R
- If it's not there, see if it's to the left or right.
- If it's left, change R.
- If it's right, change L.

8. Consider the following modification of Binary Search. Here A is a list containing distinct integers. Note that the only changes are what happens after the **while** loop ends.

```
function BINARYSEARCH(A,TARGET)
    L = 0
    R = len(A) - 1
    while L <= R do
        C = floor((L+R)/2)
        if TARGET = A[C] then
            return C
        end if
        if TARGET < A[C] then
            R = C - 1
        end if
        if TARGET > A[C] then
            L = C + 1
        end if
    end while
    if L <= len(A) - 1 then
        return L
    else
        return False
    end if
end function
```

(a) What is returned for the list `[1,3,5,7]` with `TARGET = 5`? [3 pts]

Solution:

Returns 2.

(b) What is returned for the list `[1,3,5,7]` with `TARGET = 4`? [3 pts]

Solution:

Returns 2.

(c) What is returned for the list `[1,3,5,7]` with `TARGET = 8`? [4 pts]

Solution:

Returns False.

(d) How is this algorithm different from regular Binary Search? [5 pts]

Solution:

If the number is not in the list it returns the index of the next highest number, or False if that's not possible.

9. Extra Credit: Fill in the speech bubbles with something witty. [≤ 3 pts]