CSCI-4470 Algorithms

CSCI4470 Midterm

Midterm

CSCI-4470 Midterm

Note: Show all you work and write all your answers on a separate white sheet of paper and at the end submit both your answer sheet and question paper.

1. (2 points) (True/False) Worst case runtime of SELECT algorithm (Anding Ith order statistics) is 0(n*).

> **True**

2. (2 points) $f(n) = 3n^3 - 4n^2 + 10$ **and** $h(n) = n^2 + 10n - 7$ **which of the following is correct.**

(a) $f(n) \in O(n^3)$ and $h(n) \in \omega(n^2)$
(b) $f(n) \in o(n^3)$ and $h(n) \in \Omega(n^2)$
(c) $f(n)$ and $o(n) \in o(n^3)$
(d) $f(n) \in \Omega(n^3)$ **and** $h(n) \in O(n^2)$

3. (2 points) Consider the following recurrence relation. $T(n) = T(\frac{n}{10}) + T(\frac{9}{10}) + \Theta(n)$
What is the maximum height of the recurrence tree.
(a) $\log_{10} n$
(b) $\log_{\frac{10}{9}} n$
(c) $\log_{\frac{n}{10}}(n)$
(d) $\log_{\frac{1}{10}}(n)$

4.

(a)
(b) $\frac{1}{\log(n)}$
(c) $\frac{1}{n}$
(d) $\frac{1}{n!}$

5. (2 points) What is the smallest possible depth of a leaf in a decision tree for a comparison sort algorithm? Assume the size of input is n.
(a) $\log(n)$
(b) $n-1$
(c) n^2
(d) $n\log(n)$

> To find the smallest possible depth of a lear in the decision tree.

6. (3 points) Which of the following is/are true for the sorting and selection algorithms. Select all that apply.

(a) Worst case of merge sort is $O(n\log(n))$
(b) Best case of heap sort is $O(n)$
(c) Best case of insertion sort is $O(n)$
(d) Average case of Randomized select algorithm is $O(n\log(n))$
(e) Average case of quick sort algorithm is $O(n\log(n))$

7. (2 points) Consider the following array: `A < 15, 33, 25, 28, 36, 22, 19, 24, 17, 30, 26, 32, 27, 45, 11 >`
How many nodes violate the max heap property in A because of the values of their children?
(a) One
(b) Three
(c) Two
(d) Four or more

8. (4 points) Draw the heap after fixing the nodes that violate the `max-heap` **property in question 7.**

9. (4 points) What is function `PARTITION` **going to return (give the index number) when it is called on the array** `A = (3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)` **. What will be the complexity of quick sort function on this data. Provide an explanations to your answer.**

10. (2 points) We know that in a `max-heap` **the largest number is located at the root. Where in a** `max-heap` **might the smallest element reside, assuming that all elements are distinct?**

11. (4 points) The table below is a selection table for Matrix Chain Mul-tiply. The table is not turned sideways as the book does it. The rows are the i values and the columns are the j values. What is the optimal parenthesizing for this matrix chain?

i/j	1	2	3	4	5	6
1	0	1	1	3	1	3
2		0	2	3	3	3
3			0	3	3	3
4				0	4	5
5					0	5
6						0

Determine the optimal parenthesizing for the matrix chain using the table provided.

11.1 Understand the Table:
The table is a representation of the optimal split (k value) for the matrix chain multiplication. The value in the cell (i, j) indicates where to split the matrix chain $A[i...j]$ for optimal multiplication.

11.2 Start with the Entire Chain:
We want to determine the optimal parenthesizing for the entire chain $A[1...6]$. According to the table, the value at $(1, 6)$ ***is*** 3. This means the optimal split for $A[1...6]$ is between $A[3]$ and $A[4]$.

$$(A_1 \times A_2 \times A_3) \times ...$$

11.3 Break Down Further:
Now, we break the problem into two subproblems: $A[1...3]$ and $A[4...6]$.

For $A[1...3]$, the table value at $(1, 3)$ ***is*** 1. This means the optimal split for $A[1...3]$ is between $A[1]$ and $A[2]$.

$$(A_1) \times (A_2 \times A_3)$$

For $A[4...6]$, the table value at $(4, 6)$ ***is*** 5. This means the optimal split for $A[4...6]$ is between $A[5]$ and $A[6]$.

$$(A_4 \times A_5) \times A_6$$

11.4 Construct the Parenthesizing:
Using the splits determined:

$$A[1...6] = ((A[1]) \times (A[2] \times A[3])) \times ((A[4] \times A[5]) \times A[6])$$

So, the optimal parenthesizing for the matrix chain is:

$$((A_1) \times (A_2 \times A_3)) \times ((A_4 \times A_5) \times A_6)$$

12. (2 points each) The last loop in the Counting-Sort algorithm goes from $j : n$ downto 1. What will happen if loop goes from $j : 1$ downto n.

(a) (Yes or No) Is Counting-Sort algorithm still going to sort the numbers correctly?

> **Yes**

(b) (Yes or No) Is Counting-Sort algorithm still going to be a stable algorithm?

> **No**

13. (3 points) Use the Big O definition to prove or disprove the following. $min\{f(n), g(n)\} \in O(f(n) + g(n))$
You can assume that functions involved are asymptotically non-negative functions.

Proof for $min\{f(n)\}$:

1. **Using the Definition**:
   Given $min\{f(n)\}$, for any n, this value will always be less than or equal to $f(n)$. So, $min\{f(n)\} \leq f(n)$.
2. **Combining with** $g(n)$:
   Adding $g(n)$ to both sides of the inequality, we get:
   $$min\{f(n)\} + g(n) \leq f(n) + g(n)$$
3. **Final Statement for** $min\{f(n)\}$:
   From the above inequality, it's evident that $min\{f(n)\}$ is bounded above by $f(n) + g(n)$. Thus, by the definition of Big O:
   $$min\{f(n)\} \in O(f(n) + g(n))$$

Proof for $min\{g(n)\}$:

The proof for $min\{g(n)\}$ is analogous to the proof for $min\{f(n)\}$. By the same logic:

1. $min\{g(n)\} \leq g(n)$
2. Adding $f(n)$ to both sides: $min\{g(n)\} + f(n) \leq f(n) + g(n)$
3. Thus, $min\{g(n)\} \in O(f(n) + g(n))$

Both $min\{f(n)\}$ and $min\{g(n)\}$ are individually in $O(f(n) + g(n))$.

14. (2 points each) The table below is a selection table for the knapsack problem.

X_i	X_1	X_2	X_3	X_4
W_i	3	3	2	2
V_i	14	8	6	7

(a) Looking at the table below define the knapsack problem. You can define the problem in words or you can also give an objective function and constraint for the problem.

Objective Function:

$$max\left(\sum_{i=1}^{4}\right)V_i$$

$$\sum_{i=1}^{4}W_iX_i \leq 9$$

(b) What is the optimal solution for this problem in terms of weights and value.

$W_i = X_1, X_2, X_3$
$14, 8, 7$