CECS 528, Exam 1, Fall 2024, Dr. Ebert

Unit 1 LO Problems (25 pts each)

LO1. Solve the following problems.

(a) Use the Master Theorem to determine the growth of $T(n)$ if it satisfies the recurrence $T(n) = 4T(n/2) + n^2 \log^2 n$. Defend your answer. (10 pts)

$n^{\log 4} = n^2 \Rightarrow$ By Case 4 of M.T.

$\boxed{T(n) = \Theta(n^2 \log^3 n)}$

(b) Use the substitution method to prove that, if $T(n)$ satisfies

$$T(n) = 4T(n/2) - n \log n,$$

then $T(n) = \Omega(n^2)$. (15 pts)

Assume $T(k) \geq ck^2 + dk\log k$ for all $k < n$ and some constants $c > 0$ and d.

show $T(n) \geq cn^2 + dn\log n$.

$T(n) = 4T(n/2) - n\log n \geq 4c\left(\frac{n}{2}\right)^2 + 4\left[d\left(\frac{n}{2}\right)\log\left(\frac{n}{2}\right)\right]$

$- n\log n = cn^2 + 2dn(\log n - 1) - n\log n$

$= \boxed{cn^2} + 2dn\log n - n\log n - 2dn \geq \boxed{cn^2} \iff$

$d(2n\log n - 2n) \geq n\log n \iff$

$d \geq \dfrac{n\log n}{2n\log n - 2n} = \dfrac{1}{2 - \frac{2}{\log n}}$

which is true for $d \geq 1$ and n suff. large (say $n \geq 3$).

LO2. Solve the following problems.

(a) Given that $r = ae + bg$, $s = af + bh$, $t = ce + dg$, and $u = cf + dh$ are the four entries of AB, and Strassen's products are obtained from matrices

$$A_1 = a, B_1 = f - h, A_2 = a + b, B_2 = h, A_3 = c + d, B_3 = e, A_4 = d, B_4 = g - e,$$

$$A_5 = a + d, B_5 = e + h, A_6 = b - d, B_6 = g + h, A_7 = a - c, B_7 = e + f,$$

Compute $P_1, \ldots, P_7$ and use them to compute r, s, t, and u. (10 pts)

See Divide and Conquer Lecture notes (annotated)

(b) Recall that the `Minimum Positive Subsequence Sum (MPSS)` problem admits a divide-and-conquer algorithm that, on input integer array a, requires computing the mpss of any subarray of a that contains both $a[n/2-1]$ and $a[n/2]$ (the end of a_{left} and the beginning of a_{right}. For

$$a = 48, -37, 32, -33, 70, -64, 46, -34, 45, -72, 34, -52$$

i. Provide the two sorted arrays a and b from which the minimum positive sum $a[i]+b[j]$ represents the desired mpss, for some i in the index range of a and some j within the index range of b. (8 pts)

Left Sums = -64, 6, -27, 5, -32, 16

Right Sums = 46, 12, 57, -15, 19, -33

a = sorted Left Sums = -64, -32, -27, 5, 6, 16

ii. For the a and b in part a, demonstrate the O(n) procedure that obtains the minimum positive sum $a[i] + b[j]$. Make a table that shows each of the steps. (7 pts)

b = Sorted Right Sums = -33, -15, 12, 19, 46, 57

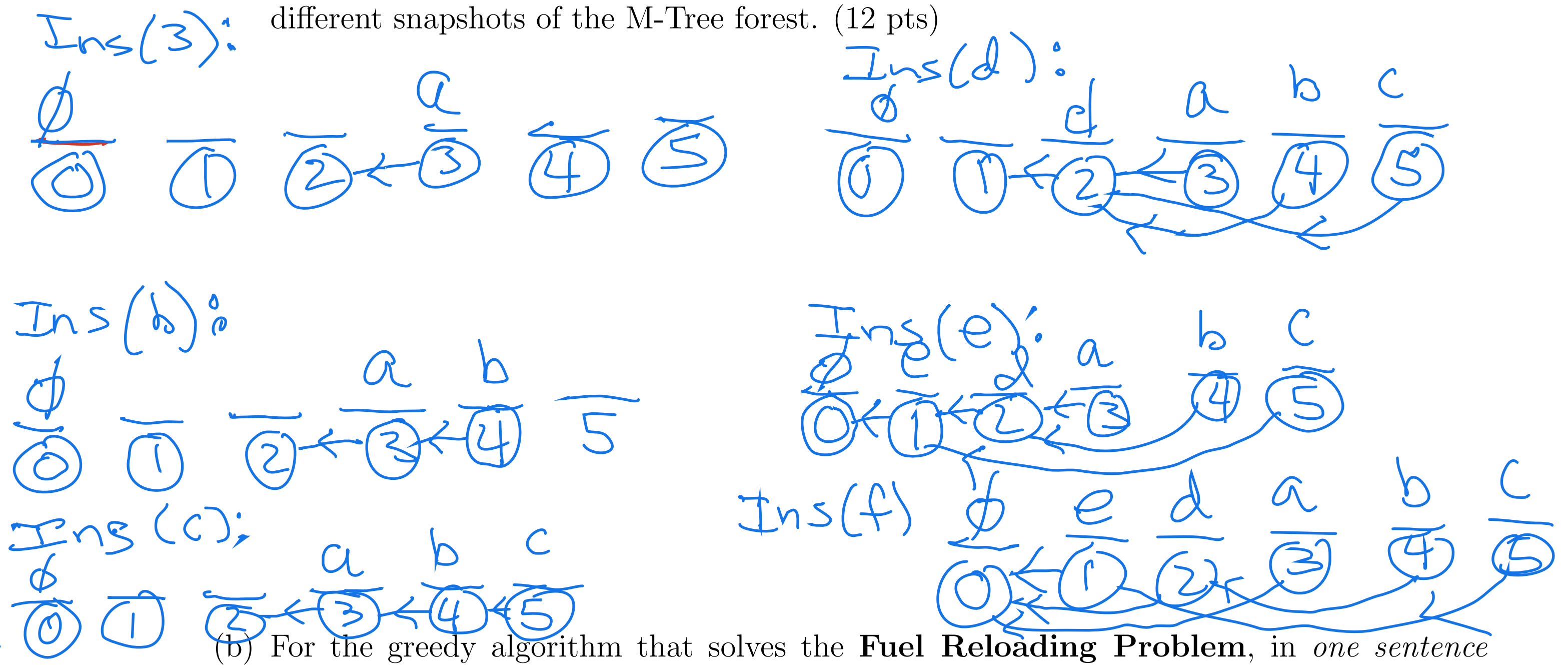
i	j	a[i] + b[j]
0	5	-7
1	5	25
1	4	14
1	3	-13
2	3	-8
3	3	24
3	2	17
3	1	-10
4	1	-9
5	1	1
5	0	-17

LO3. Do the following.

(a) Recall the use of the disjoint-set data structure for the purpose of improving the running time of the `Unit Task Scheduling` algorithm. For the set of tasks

Task	a	b	c	d	e	f
Deadline Index	3	4	5	5	5	3
Profit	60	50	40	30	20	10

For each task, show the M-Tree forest after it has been inserted (or at least has attempted to be inserted in case the scheduling array is full). Note that the earliest possible deadline index is 1, meaning that the earliest slot in the schedule array has index 1. Also, assume that an insert attempt that takes place at index i results in the function call `root`(i), followed by a `union` operation. Finally, to receive credit, your solution should show six different snapshots of the M-Tree forest. (12 pts)



(b) For the greedy algorithm that solves the **Fuel Reloading Problem**, in *one sentence* describe the greedy choice that is made in each round of the algorithm. Do *not* provide an entire description of the algorithm. (5 pts)

Choose the next station to be the one furthest from the current one, yet still reachable from the current one on a full tank of fuel.

(c) Given the station locations

$$10, 13, 19, 23, 26, 32, 36, 47, 56, 66, 73, 77, 89, 98,$$

determine the *least* distance d that a car needs to be able to travel on a full tank of fuel in order to be able to reach location 100 starting from location 0. For this distance, provide a minimum set of stations that it must visit. (8 pts)

$d = 12$ Min Set $= \{10, 19, 26, 36, 47, 56, 66, 77, 89\}$

LO4. Do the following.

(a) The dynamic-programming algorithm that solves the `Optimal Binary Search Tree (OBST)` problem defines a recurrence for the function wac(i, j). Provide the dynamic-programming recurrence for wac(i, j). (5 pts)

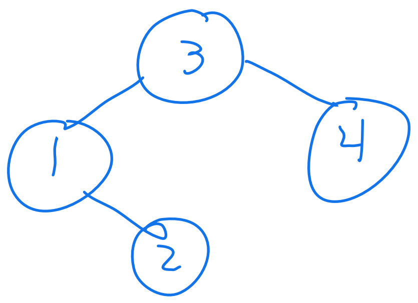
See Lecture Notes

(b) Apply the recurrence from Part b to the keys 1-4 whose respective weights are 80,40,70,50 Show the matrix of subproblem solutions and use it to provide an optimal binary search tree. For each subproblem solution, make sure to indicate the value of k that produced the solution. (13 pts)

Wac

i \ j	1	2	3	4
1	80	160 $\boxed{k=1}$	340 $\boxed{k=1}$	550 $\boxed{k=3}$
2	0	40	150 $\boxed{k=3}$	250 $\boxed{k=3}$
3	0	0	70	170 $\boxed{k=3}$
4	0	0	0	50

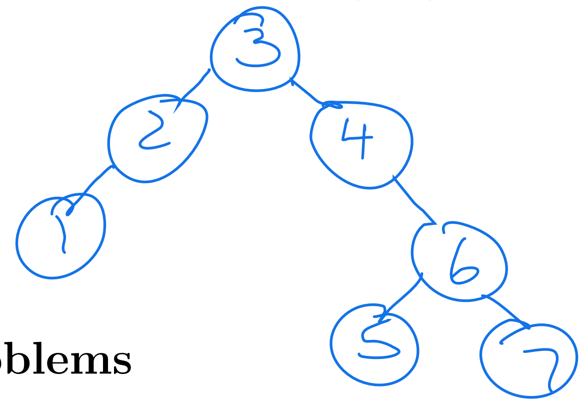
Optimal BST:



(c) When performing the dynamic-programming algorithm for an instance of `OBST` having $n = 7$ keys, the algorithm produced a matrix of k values equal to

$$\begin{pmatrix} - & 2 & 3 & 2 & 4 & 1 & 3 \\ - & - & 3 & 2 & 3 & 6 & 5 \\ - & - & - & 3 & 5 & 4 & 7 \\ - & - & - & - & 7 & 5 & 4 \\ - & - & - & - & - & 5 & 6 \\ - & - & - & - & - & - & 6 \\ - & - & - & - & - & - & - \end{pmatrix}$$

Draw the optimal binary search tree. (7 pts)



Advanced Problems

A1. Solve the recurrence $T(n) = 4T(\sqrt[3]{n}) + \log^2 n$. Hint: Let $S(k) = T(\overset{2^k}{\cancel{3^k}})$ and write a divide-and-conquer recurrence for $S(k)$. Your final answer should be expressed in terms of n. (35 pts)

$S(k) = T(2^k) = 4T(2^{k/3}) + k^2 =$

$4S(k/3) + k^2$. $k^2 = \Omega(k^{\log_3 4 + \varepsilon})$

for $\varepsilon = 2 - \log_3 4$. $\therefore$ By Case 3 of the

M. T., $S(k) = \Theta(k^2) = \boxed{\Theta(\log^2 n)}$

A2. An instance of the problem `Min Split` is a nonnegative integer array a and a positive integer k. The problem is to determine how to split the array into k subarrays so that the largest sum of any subarray is minimized. For example, if $a = 3, 4, 1, 5, 4, 2, 5, 9$ and $k = 3$, then a minimal split is $a_1 = 3, 4, 1, 5$, $a_2 = 4, 2, 5$, and $a_3 = 9$ which has a maximum sum of 13.

(a) Provide a one or two paragraph description of what you believe to be the best efficient algorithm for solving this problem and demonstrate your algorithm on the instance $a = \boxed{5, 14, 4, 3}\boxed{11, 8, 10}\boxed{3, 9, 12}$ and $k = 3$. (15 pts)

Use dynamic programing.

$mls(i,j)$ is the min. largest sum when dividing $a[i:n-1]$ into j subarrays, $0 \le i < n$, $1 \le j \le k$

Recurrence:

$$mls(i,j) = \begin{cases} M & \text{if } j = 1 \quad \text{where } M = \sum_{r=0}^{n-1} a[r] \\ a[i] & \text{if } i = n-1 \\ \min\limits_{i \le s < n-1} \left\{ \max\left(\sum_{r=i}^{s} a[r], mls(s+1, j-1) \right) \right\} \end{cases}$$

i \ j	1	2	3
0	79	42	29
1	74	40	29
2	60	34	21
3	56	32	21
4	53	29	21
5	42	21	18
6	34	21	12
7	24	12	12
8	21	12	12
9	12	12	12

$mls(2,2) = \min(56, 53, 42, 34) = 34$

$mls(3,3) = \min(29, 21, 22) = 21$

$mls(3,2) = \min(53, 42, 34, 32) = 32$

$mls(4,3) = \min(21, 21) = 21$

$mls(2,3) = \min(32, 29, 21) = 21$

$mls(5,2) = \min(34, 24, 21) = 21$

$mls(5,3) = \min(21, 18) = 18$

$mls(4,2) = \min(42, 34, 29) = 29$

$mls(1,2) = \min(60, 56, 53, 42, 40)$

$mls(1,3) = \min(34, 32, 29, 32)$

$mls(0,2) = \min(74, 60, 56, 53, 42) = 42$

$mls(0,3) = \min(40, 34, 32, 29) = \boxed{29}$

(b) Provide a convincing argument that your algorithm will always return the optimal split. (15 pts)

The dynamic-programming recurrence minimizes over the maximum sum obtained when the first array in the partition has $s-i+1$ members.

(c) Analyze the worst-case running time of your algorithm. (5 pts)

Size parameter $= n$ (assuming $O(1)$ addition).

Note that $K \leq n$.

$O(n^2)$ additions per column and K columns $\Rightarrow O(n^3)$