CECS 528, Learning Outcome Assessment 7, Spring 2024, Dr. Ebert

NO NOTES, BOOKS, ELECTRONIC DEVICES, OR INTERPERSONAL COMMUNICATION ALLOWED. Submit each solution on a separate sheet of paper.

Problems

LO3. Solve the following problems.

(a) Recall that, for the `Randomized Quicksort Algorithm`, $T(n)$ denotes the expected running time of the algorithm when applied to an array a of distinct integers where size$(a) = n$. Provide an equation for $T(n)$ conditioned on the event that the pivot M, selected for the root level of recursion, is such that there are 25 members of a that are less than M. Explain why the right side of the equation must equal $T(n)$. Hint: your answer should include discussion of how to compute the expectation of a sum of random variables.

Solution. The recurrence is

$$T(n) = T(25) + T(n - 26) + O(n).$$

This is true since, after choosing the initial pivot, the remaining steps are to i) perform the patitioning step at the root level which takes $O(n)$ steps, ii) sort the left array of size 25, and iii) sort the right array of size $n - 26$. Furthermore, the expected number of steps for each is respectively $O(n)$, $T(25)$, and $T(n - 26)$. Finally, $T(n)$ is the sum of these expectations since the expectation of a sum is the sum of the expectations. □

(b) Consider the following algorithm of Karatsuba called `multiply` for multiplying two n-bit binary numbers x and y. In what follows, we assume n is even. Let x_L and x_R be the leftmost $n/2$ and rightmost $n/2$ bits of x respectively. Define y_L and y_R similarly. Let P_1 be the result of calling `multiply` on inputs x_L and y_L, P_2 be the result of calling `multiply` on inputs x_R and y_R, and P_3 the result of calling `multiply` on inputs $x_L + x_R$ and $y_L + y_R$. Then return the value $P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2$. Apply this algorithm to the numbers $x = 27$ and $y = 36$. Only show the top level of the recursion (i.e. do *not* make a recursion tree) and the subproblem solutions that are needed to compute the final product.

Solution.

$x = \underline{0}\ \underline{1}\ \underline{1}\ \underline{0}\ \underline{1}\ \underline{1}$ $\qquad y = \underline{1}\ \underline{0}\ \underline{0}\ \underline{1}\ \underline{0}\ \underline{0}$

$x_L = 011 = (3)_2 \quad x_R = 011 = (3)_2 \qquad y_L = 100 = (4)_2 \quad y_R = 100 = (4)_2$

$P_1 = 12 \quad P_2 = 12 \quad P_3 = 6 \cdot 8 = 48$

$xy = 12 \cdot 64 + (48 - 12 - 12) \cdot 8 + 12 = 972.$

$$\begin{array}{r} 36 \\ 27 \\ \hline 252 \\ 720 \\ \hline 972 \end{array}$$

LO4. Answer the following.

(a) The FFT algorithm owes its existence to what two properties that are possessed by the nth roots of unity when n is even? See Lecture Notes (Prop. 2.7 of FFT Lecture)

(b) Compute $\mathrm{DFT}_4^{-1}(4, -3, 1, 7)$ using the IFFT algorithm. Show the solution to each of the seven subproblem instances and, for each one, clearly represent it using DFT^{-1} notation and apply the formula for computing it. Show all work.

Solution.

$\mathrm{DFT}_4^{-1}(4,-3,1,7) = \frac{1}{2}\left[\left(\frac{5}{2}, \frac{3}{2}, \frac{5}{2}, \frac{3}{2}\right) + (1,-i,-1,i) \odot (2,-5,2,-5)\right] = \left(9/4, \frac{3}{4} + \frac{5}{2}i, \frac{1}{4}, \frac{3}{4} - \frac{5i}{2}\right)$

$\mathrm{DFT}_2^{-1}(4,1) = \frac{1}{2}\left[(4,4) + (1,-1) \odot (1,1)\right] = \left(\frac{5}{2}, \frac{3}{2}\right)$

$\mathrm{DFT}_2^{-1}(-3,7) = \frac{1}{2}\left[(-3,-3) + (1,-1) \odot (7,7)\right] = \boxed{(2,-5)}$

$\mathrm{DFT}_1^{-1}(4) = \boxed{4}$ $\mathrm{DFT}_1^{-1}(1) = \boxed{1}$

$\mathrm{DFT}_1^{-1}(-3) = \boxed{-3}$ $\mathrm{DFT}_1^{-1}(7) = \boxed{7}$

LO5. For the weighted graph with edges

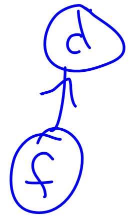
$$(a, c, 5), (b, c, 4), (c, e, 6), (c, f, 3), (c, d, 2), (d, f, 1),$$

Show how the forest of M-Trees changes when processing each edge in Kruskal's sorted list of edges. When unioning two trees, use the convention that the root of the union is the root which has the *lower* alphabetical order. For example, if two trees, one with root a, the other with root b, are to be unioned, then the unioned tree should have root a.

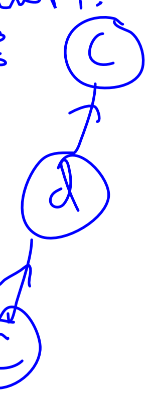
Note: only non-singleton trees are drawn.

Solution.

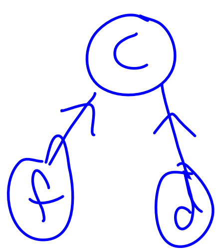
1. $(d,f,1)$:



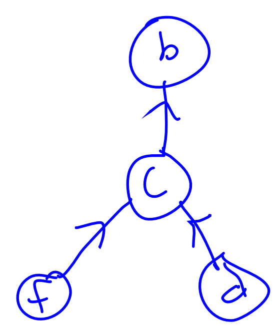
2. $(c,d,2)$:



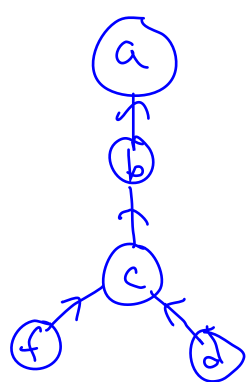
3. $(c,f,3)$:



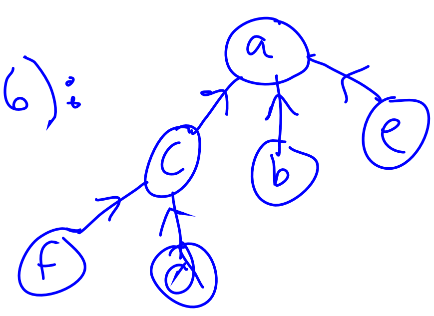
4. $(b,c,4)$:



5: $(a,c,5)$:



6. $(c,e,6)$:



LO6. Answer the following with regards to a correctness-proof outline for Kruskal's algorithm.

(a) In the correctness proof of Kruskal's algorithm, suppose $T = e_1, \ldots, e_{n-1}$ are the edges selected by Kruskal's algorithm (in that order) and T_{opt} is an mst that has edges $e_1, \ldots, e_{k-1}$, but for which $e_k \notin T_{\text{opt}}$. Then $T_{\text{opt}} + e_k$ has a cycle C. Explain why C cannot contain any edges that were rejected by Kruskal before the round for which e_k was added to the tree.

Solution. Since the edges rejected by Kruskal before the round for which e_k is added are rejected because each makes a cycle with edges $e_1, \ldots, e_{k-1}$, and since $e_1, \ldots, e_{k-1} \in T_{\text{opt}}$, it follows that T_{opt} also cannot have any of these rejected edges and thus neither can C.

(b) Explain why the fact that was stated in part a implies that C must have at least one edge e that comes after e_k in the Kruskal order.

Solution. Since C cannot possess any of the rejected edges, it must contain an edge e that comes after e_k in the sorted order. Otherwise, C would possess at most $e_1, \ldots, e_k$ which do not form a cycle (why?).

(c) Prove that $T_{\text{opt}} - e + e_k$ an mst?

Solution. Since e comes after e_k in the Kruskal sorted order, we must have $w(e) \geq w(e_k)$. Also, since $T_{\text{opt}} + e_k$ is a connected graph that has exactly one cycle of which both e_k and e are members, we see that $T_{\text{opt}} - e + e_k$ is acyclic, yet remains connected. Finally, it's an mst since e is being substituted with e_k and $w(e_k) \leq w(e)$.

LO7. Solve the following problems.

(a) The dynamic-programming algorithm that solves the `Longest Common Subsequence (LCS)` optimization problem defines a recurrence for the function lcs(i, j). In words, what does lcs(i, j) equal? Hint: do *not* write the recurrence (see Part b).

Solution. lcs(i, j) represents the longest common subsequence between the word prefixes $u[1 : i]$ and $v[1 : j]$

(b) Provide the dynamic-programming recurrence for lcs(i, j).

Solution.

$$\text{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \max(\text{lcs}(i-1,j), \text{lcs}(i,j-1)) & \text{if } u_i \neq v_j \\ \text{lcs}(i-1,j-1)+1 & \text{otherwise} \end{cases}$$

(c) Apply the recurrence from Part b to the words u = aabbab and v = bababa. Show the matrix of subproblem solutions and use it to provide an optimal solution.

Solution.

lcs(i,j)

i \ j	λ	b	a	b	a	b	a
λ	0	0	←0	←0	←0	←0	←0
a	0	↑0	↖1	←1	↖1	←1	↖1
a	0	←0	↖1	←1	↖2	←2	↖2
b	0	↖1	←1	↖2	↑←2	↖3	←3
b	0	↖1	↑←1	↖2	↑←2	↖3	↑←3
a	0	↑1	↖2	↑←2	↖3	↖3	↖4
b	0	↑1	↑2	↖3	↑←3	↖4	↑←4

abab is a longest LCS.