CECS 528, LO Assessment 1, TuTh Fall 2026, Dr. Ebert

IMPORTANT: For each problem (LO1,LO2,GK,Advanced, All LO Makeups) write your solution using a **SINGLE** and **SEPARATE SHEET OF PAPER (FRONT AND BACK)**. Write your **FIRST NAME**, **ALPHABETIZING NAME** and **PROBLEM NUMBER** on each sheet.

25 Points Each

LO1 a. Use the Master Theorem to determine the growth of $T(n)$ if it satisfies the recurrence $T(n) = 27T(n/3) + 500n^3 \log^2 n + n^2$. Defend your answer. (10 pts)

Solution. $f(n) = 500n^3 \log^2 n + n^2 = \Theta(n^3 \log^2 n)$. Similarly $n^{\log_3 27} = n^3 = \Theta(n^3)$. Thus, by Case 4 of MT, $T(n) = \Theta(n^3 \log^3 n)$.

b. Use the Substitution method to establish that, if $T(n) = 3T(n/3) + T(n/2) + n^2$, then $T(n) = \mathrm{O}(n^2)$. (15 pts)

Solution. Inductive assumption: $T(k) \leq Ck^2$ for all $k < n$ and some constant $C > 0$. Prove that $T(n) \leq Cn^2$. We have

$$T(n) = 3T(n/3) + T(n/2) + n^2 \leq 3C(n/3)^2 + C(n/2)^2 + n^2 = 7Cn^2/12 + n^2 \leq Cn^2 \Leftrightarrow$$

$$5C/12 \geq 1 \Leftrightarrow C \geq \frac{12}{5}.$$

LO2 a. Consider Karatsuba's algorithm for multiplying two n-bit binary numbers x and y, where we assume that 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 $P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2$. Demonstrate Karatsuba's algorithm on $x = 34$ and $y = 41$, where both number are assumed to require six bits. Limit your demonstration to the root level of recursion. In other words, do not go deeper than level 0. (12 pts)

Solution. $x = 100010$, $x_L = 100$, $x_R = 010$, $y = 101001$, $y_L = 101$, $y_R = 001$. Hence, $P_1 = 20$, $P_2 = 2$, and $P_3 = 36$ which yields

$$xy = 20 \cdot 64 + 14 \cdot 8 + 2 = 1394$$

which is the desired result.

b. For the `Minimum Positive Subsequence Sum` problem answer the following questions.

(a) Describe the "divide" part of the algorithm. What are the subproblems that are recursively solved? (4 pts)
Solution. See Exercise 11 of the Divide and Conquer lecture.

(b) Describe all the steps that must be taken in the "combine" portion of the algorithm, including how the subproblem solutions are utilized and the big-O number of total steps $f(n)$ that are required by the combine portion. Please keep your description as general as possible and *do not* work through an example. (9 pts)
Solution. See Exercise 11 of the Divide and Conquer lecture.

GK a. Using just a few sentences, why is the Master Theorem by itself insufficient for determining the big-O number of steps that is needed to execute any arbitrary divide-and-conquer algorithm? (10 pts)

Solution. Some divide-and-conquer algorithms, such as `Find Statistic`, yield non-uniform recurrences for which the Master Theorem does not apply.

b. For a uniform divide-and-conquer recurrence, what do the a, b, and $f(n)$ represent assuming that $T(n)$ is describing the number of steps that are needed to execute some divide-and-conquer algorithm? (15 pts)

Solution. See Divide and Conquer lecture notes.

ADV Solve the recurrence $T(n) = 4T(\sqrt{n}) + \log^2 n \log(\log n)$. Hint: Let $S(k) = T(2^k)$ and use this identity to provide a recurrence for $S(k)$. (25 pts)

Solution. Since k represents $\log n$, the $T(n)$ recurrence translates to $S(k)$ as

$$S(k) = 4S(k/2) + k^2 \log k$$

which implies by Case 4 of the Master Theorem that $S(k) = \Theta(k^2 \log^2 k)$. Therefore, $T(n) = \Theta(\log^2 n \log^2(\log n)$.