# CECS 528, Learning Outcome Assessment 7a, March 17th, Spring 2023, Dr. Ebert

**NO NOTES, BOOKS, ELECTRONIC DEVICES, OR INTERPERSONAL COMMUNICATION ALLOWED. Submit solutions to at most 2 LO problems on separate sheets of paper**.

## Problems

LO3. The following pertains to a correctness-proof outline for the Unit Task Scheduling (UTS) algorithm. Let $S = (a_1, t_1), \ldots, (a_m, t_m)$ represent the tasks that were selected by the algorithm for scheduling, where $a_i$ is the task, and $t_i$ is the time that it is scheduled to be completed, $i = 1, \ldots, m$. Moreover, assume that these tasks are ordered in the same order for which they appear in the sorted order. Let $S_{\text{opt}}$ be an optimal schedule which also consists of task-schedule-time pairs. Let $k$ be the first integer for which $(a_1, t_1), \ldots, (a_{k-1}, t_{k-1})$ are in $S_{\text{opt}}$, but $(a_k, t_k) \notin S_{\text{opt}}$ because $a_k$ was not scheduled by $S_{\text{opt}}$.

(a) How do we know that $S_{\text{opt}}$ must have a task $a$ scheduled at $t_k$? Hint: what contradiction arises in case time $t_k$ is not being utilized?

(b) What contradiction arises when we assume that $a$ comes before $a_k$ in the UTS ordering? Hint: there are two cases.

LO4. Given $T_1(n) = 16T_1(n/4) + n^2$, and $T_2(n) = aT_2(n/3) + n^2$ what is the greatest possible value that $a$ can assume, and still have $T_2(n) = o(T_1(n))$? Show work and explain.

LO5. When applying the linear-time `Merge` algorithm used by Mergesort to the two sorted arrays $a = (1, 3, 10, 13, 20)$ and $b = (2, 4, 6, 16, 19)$, list each pair of numbers that must be compared. Hint: $(1, 2)$ is the first pair. In general, if $a$ and $b$ both have size $n$, then how many pairs of numbers must be compared?

LO6. Consider the following algorithm called `Karatsuba's algorithm` for multiplying two $n$-bit binary numbers $x$ and $y$. Let $x_L$ and $x_R$ be the leftmost $\lceil n/2 \rceil$ and rightmost $\lfloor n/2 \rfloor$ 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 $P_1 \times 2^{2\lfloor \frac{n}{2} \rfloor} + (P_3 - P_1 - P_2) \times 2^{\lfloor n/2 \rfloor} + P_2$. Show that the algorithm always works by proving in general that $xy = P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2$ for arbitrary $x$ and $y$. Hint: to avoid floors and ceilings, you may assume that $x$ and $y$ both have even lengths.

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 $\text{lcs}(i, j)$. In words, what does $\text{lcs}(i, j)$ equal? Hint: do *not* write the recurrence (see Part b).

(b) Provide the dynamic-programming recurrence for $\mathrm{lcs}(i, j)$.

(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.