CSCE 411, Fall 2025: Homework 2

Due: Friday, September 12, 11:59 pm CT

Problem 0. (-5 for leaving blank.) Write down all outside resources you consulted, and the names of any other students that you worked with in any way. By turning in this homework, you acknowledge that you have read and abide by all course policies on outside resources and collaboration as stated in the course syllabus. Use of Generative AI tools must be explicitly noted. *If you did not use outside resources, AI tools, or collaborators, state that explicitly.*

Problems 1 and 2: Abdul Bari on Youtube

Problem 3: ChatGPT

Problem 1. (10 points) Assume that you have coded up a function Strassens(A, B) that takes in two matrices A and B of size $n \times n$ where n is a power of 2, and runs Strassen's algorithm on them to compute AB in $O(n^{\log_2 7})$ time. Now consider two matrices C and D of size $m \times m$ where m is not a power of 2. Explain how you could use Strassens to compute CD in $O(m^{\log_2 7})$ time. You are allowed to manipulate the input matrices but you cannot manipulate the function itself.

Firstly, Strassen's algorithm only works when n is a power of 2, so take the size m of the matrices C and D and find the next power of 2 that is larger than m. This will be the value for n in our A and B matrices.

Now copy the entries of C into the top-left block of A and copy the entries of D into the top-left block of B. We will fill the rest of the rows and columns with zeros.

Pass A and B into Strassen's algorithm and we should get a product matrix $E = AB$. Because the extra rows and columns in A and B are all zeros, they do not contribute anything to the product.

So, $E = \begin{bmatrix} CD & 0 \\ 0 & 0 \end{bmatrix}$

Therefore, the top-left block of E is the product CD.

For time complexity, based on what we did, we know $m < n < 2m$. So, $T(n) = O((2m)^{\log_2 7}) =$

$O(2^{\log_2 7} * m^{\log_2 7}) = O(7 * m^{\log_2 7}) = O(m^{\log_2 7})$.

So asymptotically, nothing changes.

Problem 2. (20 points) You are scheduling the Aggies' upcoming football season, and your goal is to maximize the total expected attendance across all Aggie games. You must follow two rules: you cannot schedule two consecutive games at home, and you cannot skip any games. Each game must be scheduled either at home or away.

For the i-th game, if it is played at home the expected attendance is H_i, and if it is played away the expected attendance is A_i. The attendance values are given as two sequences

$$H_1, H_2, \ldots, H_n \quad \text{and} \quad A_1, A_2, \ldots, A_n,$$

where H_i and A_i denote the expected home and away attendance of the i-th game, respectively.

(a) (10 points) Given an integer t such that $1 \le t \le n$, let $A(t)$ be the maximum attendance of t games while following the rule. What is the recurrence relation for $A(t)$? Justify your answer.

$A(t) = max(A(t-1) + A_t, A(t-2) + A_{t-1} + H_t)$

Base cases: $A(1) = max(H_1, A_1)$ and $A(0) = 0$

Following the rules given, if game t is away, then the first $t-1$ games can be any valid optimal schedule for $t-1$ games. So, $A(t-1) + A_t$. If game t is home, then game $t-1$ has to be away. So, that leaves with the total to be the optimal attendance for the first $t-2$ games plus attendance for game $t-1$ away and game t home. $A(t-2) + A_{t-1} + H_t$

(b) (10 points) Write pseudo-code for an algorithm that finds the maximum attendance you can achieve while following the rules.

Algorithm 1 Max Attendance Algorithm

```
procedure MaxAttendance(n, A, H)
    ma[0] ← 0
    ma[1] ← max(H[1], A[1])
    for t = 2 to n do
        ma[t] ← max(ma[t − 1] + A[t], ma[t − 2] + A[t − 1] + H[t])
    return ma[n]
```

Problem 3. (20 points) A shipping company needs to transport goods across a chain of n cities, labeled $1, 2, \ldots, n$. The goods must start in city 1 and end in city n. For each i and j with $1 \leq i < j \leq n$, there is a direct shipping route from city i to city j with cost $c_{i,j}$, and you may assume $c_{1,1} = 1$. The company wants to minimize the total cost of transporting goods from city 1 to city n, but a shipment can only move forward (from a city with a smaller number to a city with a larger number).

(a) (10 points) Let $M(n)$ be the minimum total cost of transporting goods from city 1 to city n. Write and justify the recurrence relationship for $M(n)$.

$M_1 = 0$

$M_n = \min_{1 \leq i < n}(M_i + c_{i,n})$ up to n

The base case is $M_1 = 0$ because we don't travel anywhere.

To reach city j the final hop must come from some earlier city $i < j$. The cheapest way to reach that city is M_i, and adding the direct cost $c_{i,j}$ gives us the total cost $M_i + c_{i,j}$. Minimizing over all possible predecessors i gives the optimal cost to j, so the recurrence follows by optimal substructure.

(b) (10 points) Write pseudo-code for an algorithm that computes the minimum shipping cost $M(n)$ from city 1 to city n.

Algorithm 2 Min Shipping Cost Algorithm

procedure MinShipCost(n, c)
    $M[1..n] \leftarrow 0$
    $M[1] \leftarrow 0$
    for $j = 2$ to n **do**
        $M[j] \leftarrow \infty$
        for $i = 1$ to $j - 1$ **do**
            if $M[i] + c[i][j] < M[j]$ **then**
                $M[j] \leftarrow M[i] + c[i][j]$
    return $M[n]$