CIS4930 – Problem Solving **Spring 2026**

Homework 2: Deadline Tuesday 3/31

Instructor: Viet Tung Hoang

1. (80 points) Let $n \geq 1$ be an integer, and $N = 2^n$. Suppose that we have a floor of size $N \times N$, and I mark one particular cell of the floor. We want to tile the remaining part of the floor via triminos, as illustrated in Figure 2.1.

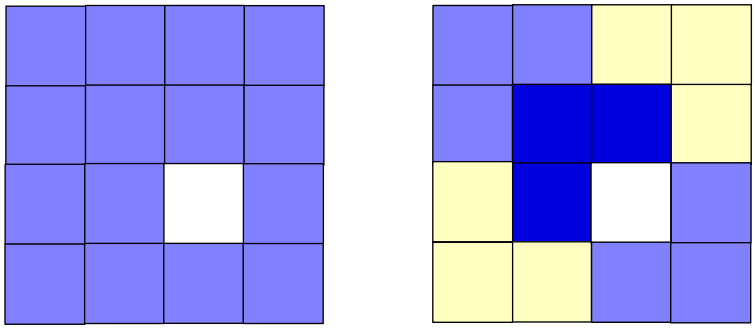

Figure 2.1: Left: A floor of size 4×4, with one marked cell. Right: How the floor on the left can be tiled via triminos.

a) (40 points) Suppose that we mark the cell at the corner of the floor. Describe a recursive algorithm to tile the floor. Your algorithm should use $O(N^2)$ steps.

b) (40 points) Now the marked cell is at an arbitrary position. Describe a recursive algorithm to tile the floor. Your algorithm should use $O(N^2)$ steps.

[**We expect an English description of your algorithm, and an informal argument for its correctness. Provide a recurrence for the running time and analyze it.**]

2. (80 points) a) (40 points) Use the substitution method, find the closed-form formula of

$$S_5(n) = 1^5 + 2^5 + \cdots + n^5 \ .$$

[**You need to describe the process that you guess the formula of $S_5(n)$. You also need to include a rigorous induction proof.**]

b) (40 points) Let

$$T(n) = \begin{cases} 1 & \text{if } n < 8 \\ T(\lfloor n/2 \rfloor) + T(\lfloor n/4 \rfloor) + T(\lfloor n/8 \rfloor) + n & \text{otherwise} \end{cases}$$

Use the substitution method, find the Big-Theta of $T(n)$.

[**You need to describe the process that you guess the formula of $T(n)$. You also need to include a rigorous induction proof.**]

3. (80 points) You're helping some security analysts monitor a collection of networked computers, tracking the spread of an online virus. There are n computers in the system, labeled $C_1, \ldots, C_n$, and as input

you're given a collection of *trace data* indicating the times at which pairs of computers communicated. Thus the data is a sequence of ordered triples (C_i, C_j, t_k); such a triple indicates that C_i and C_j exchanged bits at time t_k. There are m triples total.

We'll assume that the triples are presented to you in sorted order of time. For purposes of simplicity, we'll assume that each pair of computers communicates at most once during the interval you're observing.

The security analysts you're working with would like to be able to answer questions of the following form: If the virus was inserted into computer C_a at time x, could it possibly have infected computer C_b by time y? The mechanics of infection are simple: if an infected computer C_i communicates with an uninfected computer C_j at time t_k (in other words, if one of the triples (C_i, C_j, t_k) or (C_j, C_i, t_k) appears in the trace data), then computer C_j becomes infected as well, starting at time t_i. Infection can thus spread from one machine to another across a *sequence* of communications, provided that no step in this sequence involves a move backward in time. Thus, for example, if C_i is infected by time t_k, and the trace data contains triples (C_i, C_j, t_k) and (C_j, C_q, t_r), where $t_k \leq t_r$, then C_q will become infected via C_j. (Note that it is okay for t_k to be equal to t_r; this would mean that C_j had open connections to both C_i and C_q at the same time, and so a virus could move from C_i to C_q.)

To have a better understanding of the problem, consider the examples illustrated in Figure 2.2.

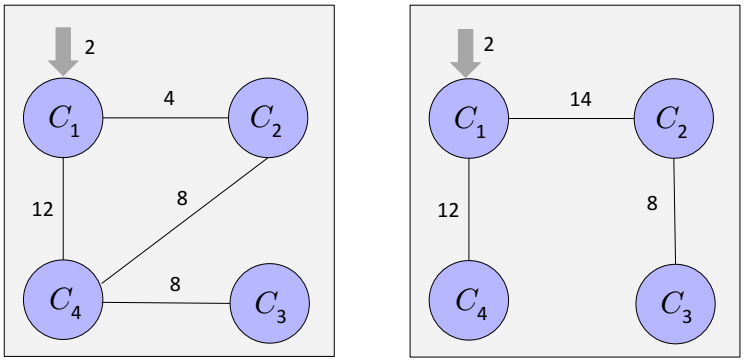

Figure 2.2: Examples of trace data for the virus-tracking problem. For a triple (C_i, C_j, t_k), we draw a directed edge from node C_i to node C_j, with label t_k. A virus was inserted to C_1 at time 2.

For the example on the left panel of Figure 2.2, machine C_3 would be infected at time 8 by a sequence of three steps: $C_1 \to C_2$ (time 4), then $C_1 \to C_4$ (time 8), and then $C_4 \to C_3$ (time 8).

For the example on the right panel of Figure 2.2, machine C_3 would not become infected during the period of observation: although C_2 becomes infected at time 14, we see that C_3 only communicated with C_2 *before* C_2 was infected. There is no sequence of communications moving forward in time by which the virus could get from C_1 to C_3 in this second example.

Design an algorithm that answers questions of this type: given a collection of trace data, the algorithm should decide whether a virus introduced at computer C_a at time x could have infected computer C_b by time y. The algorithm should run in time $O(m + n)$.

[**We expect an English description of your algorithm, and an informal argument for its correctness and running time.**]