

# Fundamental Algorithm Techniques

## Problem Set #4

No Code Required, Review on February 19

Due: February 26, 2026

**Problem 1** (Courses Allocation: Dynamic Programming VS Greedy Computing!). *For conflict free course allocation as in the course:*

*Consider set of activities:*

$$S = \{a_1, a_2, \dots, a_n\},$$

*each with starting and ending times  $s_i, f_i$ . Assume also that  $S$  is sorted such that:*

$$f_1 \leq f_2 \leq \dots \leq f_n$$

*A set of compatible activities  $\hat{S}_{ap}$  means that each activities of  $\hat{S}_{ap}$  starts after the last activity ended and ends before the next activity starts.*

- **Dynamical Approach:**

*Consider the recurrence below for the cost  $c[a, p] = c[a, k] + c[k, p] + 1$  between the course  $a_a$  and  $a_p$ :*

$$c[a, p] = \begin{cases} 0 & \text{if } \hat{S}_{ap} = \emptyset, \\ \max \{c[a, k] + c[k, p] + 1 \mid a_k \in \hat{S}_{ap}\} & \text{if } \hat{S}_{ap} \neq \emptyset, \end{cases}$$

1. *Structuring the problem with  $\hat{S}_{ap}$  actually characterises optimal substructures? Can therefore dynamical computing can be used?*
2. *Explain/draw how that can be used for dynamical programming, top down with recurrence: RecuSelect( $s, f, k, n$ ), starts ( $s, f, 0, n$ ), return list  $a_i$ 's.*
3. *dynamical programming with tabulation.*

- **Greedy Approach:**

- I. *What is the greedy choice for the activity-selection problem?*<sup>1</sup>
- II. *Write the pseudocode for the greedy approach GreedySchedule( $s, f, n$ )*  
**!!!consider sorting courses first!.**

- III. *Prove that your GreedySchedule is optimal using<sup>2</sup> (hard)*

- induction
- stay ahead arguments
- contradiction

---

<sup>1</sup>choose the activity in S with the earliest finish time and bottom up, like in the course...:-)

<sup>2</sup>See course V!