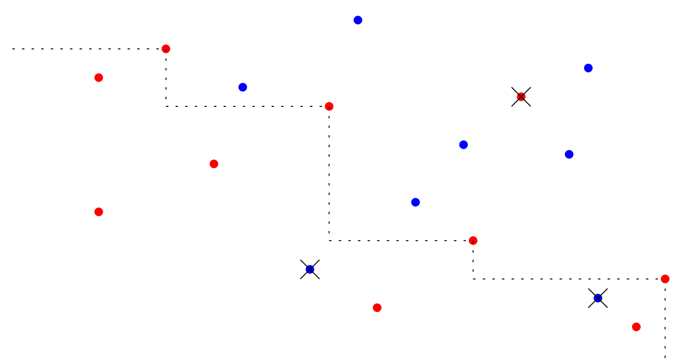
CS 473 (Fall 2026)
Homework 2 (due Sep 17 Thu 10am)

Instructions: Carefully read `http://courses.grainger.illinois.edu/cs473/fa2026/policies.html#hw` and `http://courses.grainger.illinois.edu/cs473/fa2026/integrity.html` (in particular, remember to include the 3 disclosure statements, on collaborators, outside sources, and AI use, for each problem).

Problem 2.1: A set Q of points in 2D where each point is colored red or blue is said to be *staircase-separable* if there are no red point $r = (x_r, y_r) \in Q$ and blue point $b = (x_b, y_b) \in Q$ with the red point "dominating" the blue point, i.e., with $x_r > x_b$ and $y_r > y_b$.

Consider the following problem: given a set P of n points in 2D where each point is colored red or blue, delete the smallest number of points from P so that P becomes staircase-separable.

In the example below, deleting the 3 points marked with X makes the point set staircase-separable.



Describe an efficient algorithm to solve this problem using dynamic programming. Include the following steps: (a) first define your subproblems precisely, (b) then derive the recursive formula (including base cases) with brief justifications (no need for long correctness proofs), (c) write pseudocode to output the optimal value (i.e., the minimum number of points to delete), (d) write pseudocode to output the deleted points in an optimal solution, and (e) analyze the running time and space (as a function of n).

Problem 2.2: We have n jobs, where job i must start at time t_i (these start times are all given in advance; you may assume they are sorted $t_1 < t_2 < \cdots < t_n$). We have 3 servers, each working at different speeds: for each $j \in \{1, 2, 3\}$, server j requires L_j units of time to handle each job. The goal is to choose a largest subset of jobs that can be handled by the 3 servers. Each job can only be assigned to one server. Thus, the main constraint is that whenever two different jobs i and i' are assigned to the same server j, we must have $|t_{i'} - t_i| > L_j$.

Describe an efficient (polynomial-time) algorithm to compute the optimal value (i.e., size of the largest feasible subset) using dynamic programming. Include the following steps: (a) first define your subproblems precisely, (b) then derive the recursive formula (including base cases) with brief justifications (no need for long correctness proofs), (c) specify a valid evaluation order, and (d) analyze the running time and space (as a function of n). For this problem, you do not need to write pseudocode if your recursive formula and evaluation order are described clearly. And you do not need to write pseudocode to output the optimal subset or assignment of jobs to servers.

Example: Suppose we have 6 jobs with start times $t_1 = 3$, $t_2 = 8$, $t_3 = 11$, $t_4 = 12$, $t_5 = 16$, $t_6 = 18$, and suppose we have 3 servers with $L_1 = 6$, $L_2 = 7$, $L_3 = 9$. A feasible solution is to let server 1 handle jobs 1, 3, and 6; and server 2 handle jobs 2 and 5; and server 3 handle job 4. The optimal value is 6 (all 6 jobs can be handled).

Optional Problem 2.3 (not graded):

(a) Consider a function $C(\cdot,\cdot)$ defined by the following recursive formula: for all $i, j \geq 1$,

$$C(i,j) \;=\; \max\Big\{C(i-1,j-1) + f(i,j),\; C(i+2,j-2) + g(i,j),\; C(i+j,j) + h(i,j)\Big\},$$

with the base case $C(i,j) = 0$ if $i \leq 0$ or $j \leq 0$ or $i > n$. Here, f, g, h are given functions that can be evaluated in constant time.

(i) Write pseudocode to evaluate $C(n,n)$ using dynamic programming, and (ii) analyze the running time and space as a function of n.

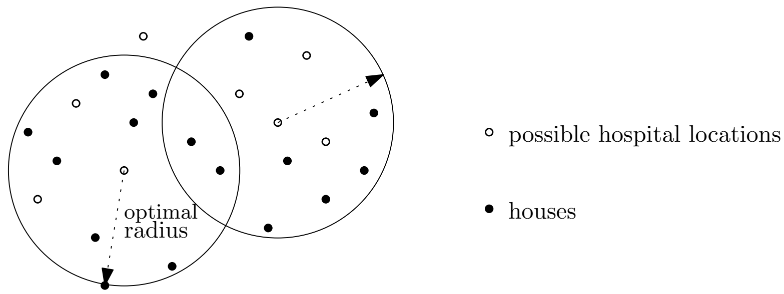
(b) Consider a function $C(\cdot,\cdot)$ defined by the following recursive formula: for all $i, j \in \{1, \ldots, n-1\}$,

$$C(i,j) \;=\; \min_{k \in \{j+1,\ldots,n\}:\; g(i,j,k) \geq 0} \Big(C(i-1,k) + C(i+1,k) + f(i,j,k)\Big),$$

with base cases $C(i,j) = 0$ if $i = 0$ or $i = n$, and $C(i,j) = 0$ if $j = n$. Here, f, g are given functions that can be evaluated in constant time.

(i) Write pseudocode to evaluate $C(\lfloor n/2 \rfloor, 1)$ using dynamic programming, and (ii) analyze the running time and space as a function of n.

Optional Problem 2.4 (not graded): We are given a list L of n locations, and a list H of n houses. For each location $p \in L$ and each house $h \in H$, we are given a distance value $d(p,h)$. We want to find two locations $p, q \in L$ to build two hospitals so that $\max_{h \in H} \min\{d(p,h), d(q,h)\}$ is as small as possible. (The quantity $\min\{d(p,h), d(q,h)\}$ represents the distance of the house h to the closer of the two hospital locations p, q. The motivation is to make the worst such distance as small as possible.)



(a) (70 pts) First give a subcubic algorithm for the following decision problem: given a value r, do there exist $p, q \in L$ such that $\max_{h \in H} \min\{d(p,h), d(q,h)\} \leq r$?

[Hint: use an algorithm from class.]

(b) (30 pts) Now give a subcubic algorithm for the original problem, using (a).