

CS 473 (Spring 2025)

Homework 10 (due May 1 Thu 10am)

Instructions: As in previous homeworks.

Problem 10.1: Consider the following problem: The input consists of n rectangles, where the i -th rectangle R_i has height $h_i \leq 1$, width $w_i \leq 1$, and area $a_i = h_i w_i$. The goal is to place the rectangles in a rectangular container of width 1 while minimizing the height of the container. Rotations of the rectangles are not allowed.

Consider the following greedy algorithm:

0. sort the rectangles so that $h_1 \geq h_2 \geq \dots \geq h_n$
1. $k_1 = 1$
2. for $m = 1, 2, \dots$ until $k_m = n + 1$ {
3. let k_{m+1} be the largest index such that $w_{k_m} + w_{k_m+1} + \dots + w_{k_{m+1}-1} \leq 1$
4. place $R_{k_m}, R_{k_m+1}, \dots, R_{k_{m+1}-1}$ in a new row (a “shelf”) of height h_{k_m}
- }

Show that this greedy algorithm runs in polynomial time and has approximation factor at most 3.

[Hint: Prove the inequality $a_{k_m} + a_{k_m+1} + \dots + a_{k_{m+1}} > h_{k_{m+1}}$. Then sum both sides...]

Problem 10.2: Consider the following variant of the traveling salesman problem: given a complete weighted graph $G = (V, E)$, where the weights satisfy $d(u, v) = d(v, u)$ and $d(u, v) \leq d(u, w) + d(w, v)$, find a cycle C that visits every vertex exactly once, minimizing the *maximum edge weight* in C . (In other words, our salesman doesn't care about the total travel time, but can't tolerate long flights and wants to keep each flight short.) Equivalently, we seek the smallest weight r^* such that $G[r^*]$ contains a Hamiltonian cycle, where $G[r]$ denotes the subgraph $(V, \{uv \in E \mid d(u, v) \leq r\})$.

Consider the following approach:

1. find the smallest r_0 such that $G[r_0]$ is connected
2. let T be any spanning tree of $G[r_0]$
3. get a (simple) cycle C from T and return C

To perform line 3, we pick an arbitrary node u as the root of T and call the following recursive procedure, which returns a simple path in G that visits all descendants of u :

- visit(u):
1. if u is a leaf then return $\langle u \rangle$
 2. let u_1, \dots, u_k be the children of u
 3. for $i = 1, \dots, k$ do $\pi_i = \text{visit}(u_i)$
 4. return $\langle u \rangle \circ (\pi_1)^R \circ (\pi_2)^R \circ \dots \circ (\pi_k)^R$

Here, \circ denotes concatenation, and π^R denotes the reverse of the list π . We get C by completing the resulting path into a cycle in G .

- (a) (70 pts) Show that the cycle C produced by the above procedure has maximum edge weight $\leq 3r_0$.
 [Hint: You may want to first deduce some properties about the path generated by `visit()`. Try a few examples and see; then use induction to prove your observations formally.]
- (b) (30 pts) Conclude that there is a polynomial-time algorithm with approximation factor at most 3.

Problem 10.3: Given a *directed* graph $G = (V, E)$, we want to find a subgraph A that is acyclic (i.e., a dag), maximizing the number of edges in A .

- (a) (50 pts) Show that the following deterministic algorithm yields a feasible solution with approximation factor at least $1/2$:
1. fix an arbitrary (not random) order of the vertices v_1, \dots, v_n
 2. for $i = 2$ to n do {
 3. $IN_i =$ all edges from $\{v_1, \dots, v_{i-1}\}$ to v_i
 4. $OUT_i =$ all edges from v_i to $\{v_1, \dots, v_{i-1}\}$
 5. if $|IN_i| \geq |OUT_i|$ then insert IN_i to A else insert OUT_i to A
 - }
- (b) (50 pts) Show that the following randomized algorithm yields a feasible solution and analyze its expected approximation factor:
1. take a random order of the vertices v_1, \dots, v_n
 2. for each edge $(v_i, v_j) \in E$ do
 3. if $i < j$ then insert (v_i, v_j) to A

[Note: it may be useful to recall the fact that a directed graph is a dag iff there exists a topological sort.]