Fall 2024 Qualifying Exam—Algorithms (750)

Question 1 (A Recurrence). **Find** tight asymptotic bounds on any positive function $T(n)$ satisfying the following recurrence for all sufficiently large n:

$$T(n) = 3T(n/2) + n\sqrt{n}\ .$$

You may assume that any implicit floors and ceilings are of no consequence.

Prove your upper bound by the substitution method. (You do not need to prove the matching lower bound, but if your upper bound is not tight, you will not receive credit even for a correct substitution proof.)

Answer: We have $3/2 < \lg 3$, so Case 1 of the Master Theorem applies so that $T(n) = \Theta(n^{\lg 3})$. (To see that $3/2 < \lg 3$ without a calculator, first take 2 to the power of both sides, then square both sides, yielding the equivalent $8 < 9$.)

Proof of the upper bound: assume $T(m) \leq cm^{\lg 3} - dm^{3/2}$ for all $m < n$, where c and d yet to be determined. Then

$$\begin{aligned} T(n) &= 3T(n/2) + n\sqrt{n} \\ &\leq 3(c(n/2)^{\lg 3} - d(n/2)^{3/2}) + n^{3/2} \\ &= cn^{\lg 3} - 3d((n/2)^{3/2} + n^{3/2} \\ &= cn^{\lg 3} - d\frac{3}{2^{3/2}}n^{3/2} + n^{3/2} \\ &= cn^{\lg 3} - dn^{3/2} - dn^{3/2}\left(\frac{3}{2\sqrt{2}} - 1\right) + n^{3/2} \\ &\leq cn^{\lg 3} - dn^{3/2} \end{aligned}$$

provided

$$\begin{aligned} n^{3/2} &\leq dn^{3/2}\left(\frac{3}{2\sqrt{2}} - 1\right) \\ 1 &\leq d\left(\frac{3}{2\sqrt{2}} - 1\right) \\ d &\geq \left(\frac{3}{2\sqrt{2}} - 1\right)^{-1} \end{aligned}$$

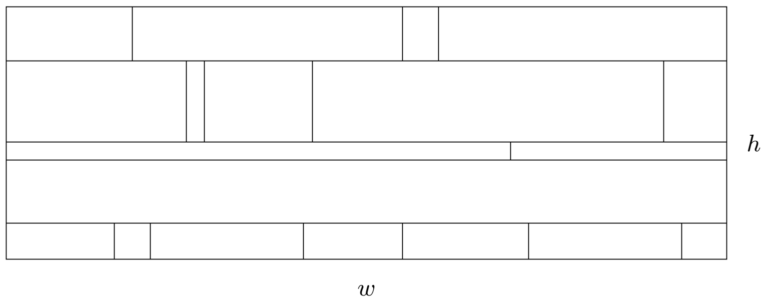
Question 2 (Optimal Restricted Sheet Cutting). You are given a sheet of metal that has width w (the horizontal dimension) and height h (the vertical dimension), where w and h are natural numbers (all units are in centimeters). You can cut the sheet and sell off the pieces. You have a table of prices $P[1..w, 1..h]$ where $P[j, k] \geq 0$ is the amount of money you can charge for a piece of width j and height k. You want to maximize the total amount you can charge selling the pieces.

You are allowed to make the following cuts in order:

- First you can make any number of horizontal cuts through the entire sheet, creating horizontal strips of various integer heights.

- You may then cut each strip into pieces with vertical cuts, independent of how you cut the other strips. The pieces must all have integer dimensions.

For example, here is legal way of cutting the sheet:



Note that cuts are not required. It may be that the best option involves no cuts at all.

Describe an algorithm that finds the maximum total price you can charge from cutting the $w \times h$ sheet. You do not need to find an optimal cut, just its total price. Describe your algorithm in enough detail so that someone who did well in CSCE 750 can implement it without specific knowledge of the problem.

For 95% credit: Your algorithm must run in time $O((wh)^2)$.

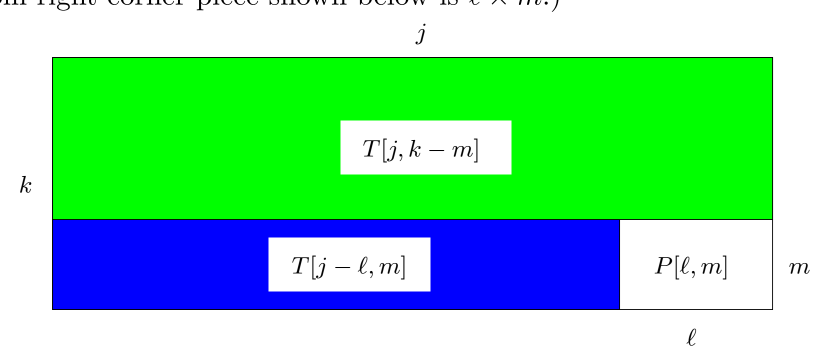
For 100% credit: Your algorithm must run in time $O(wh(w + h))$.

You can assume that all numerical arithmetic and comparison operations take $O(1)$ time each. [Hint: dynamic programming.]

Answer:

For 95% credit:

1. Allocate a table $T[0..w, 0..h]$. (Idea: $T[j,k]$ will be set to the optimal price obtained by cutting a sheet of width j and height k.)
2. Initialize $T[j,0] := T[0,k] := 0$ for all j,k. (Can't get money for nothing!)
3. For $1 \leq j \leq w$ and $1 \leq k \leq h$ (both in increasing order):
   (a) Set
   $$T[j,k] := \max_{1\leq \ell \leq j,\ 1\leq m\leq k} \{P[\ell,m] + T[j,k-m] + T[j-\ell,m]\} \ .$$
   (The sum $P[\ell,m]+T[j,k-m]+T[j-\ell,m]$ is the optimal price obtainable assuming the bottom right corner piece shown below is $\ell \times m$.)



4. Return $T[w, h]$.

For 100% credit:

1. Allocate two tables $T[0..w, 0..h]$ and $V[0..w, 1..h]$. (Idea: $T[j, k]$ is as in the 95% solution and V is like T except that it assumes no horizontal cuts. V is easier to fill in, and T is easier to fill in given V.)
2. Initialize $V[0, k] := 0$ for all $1 \le k \le h$.
3. For $1 \le j \le w$ (in increasing order) and $1 \le k \le h$ (in any order):
   (a) Set
   $$V[j, k] := \max_{1 \le \ell \le j} \{P[\ell, k] + V[j - \ell, k]\} \ .$$
4. For $1 \le j \le w$ and $1 \le k \le h$ (both in increasing order):
   (a) Set
   $$T[j, k] := \max_{1 \le m \le k} \{T[j, k - m] + V[j, m]\} \ .$$
   (The sum $T[j, k-m] + V[j, m]$ is the optimal price obtainable assuming the bottom strip has height m.)
5. Return $T[w, h]$.

Question 3 (Minimum-Weight Directed Cycle). You are given a directed graph G with weight function $w : G.E \to \mathbb{R}$ such that $w(e) \ge 0$ for all $e \in G.E$. You want to find the minimum total weight of any directed cycle passing through a given vertex s. A cycle must include at least one edge (which could be a self-loop).

Describe an algorithm that, given digraph G, weight function w, and $s \in G.V$, returns the minimum total weight of any directed cycle through s, or ∞ if there is no such cycle. You only need to return the weight, not the actual cycle. Describe your algorithm in enough detail so that someone who did well in CSCE 750 can implement it without specific knowledge of the problem.

Your algorithm should run in time $O((V + E) \lg V)$, where $V := |G.V|$ and $E := |G.E|$.

Answer: There are more than one ways of doing this. Here is probably the simplest:

1. Create a new vertex $s' \notin G.V$.
2. For every edge leaving s to a vertex v of G (including s itself), add an edge with the same weight from s' to v.
3. Run Dijkstra's algorithm on the altered graph with source s', and return $s.d$.

Explanation (optional): doing this prevents returning 0 for the empty cycle at s.

One can also make two copies of G and add a directed edge of weight 0 from each vertex other than s of the first copy to the corresponding vertex in the second copy, and if there is a self-loop ℓ at s, include an edge from s in the first copy to s in the second copy with weight $w(\ell)$. Then run Dijkstra.