Problem Set 8

Name: Gabriel Chiong

Collaborators: None

Problem 8-1.

Subproblems:

- Let $x(i, j)$ be the maximum possible profit by selling j barrels to the suffix of buyers $i, \ldots, n$ for $i \in \{1, \ldots, n+1\}$ and $j \in \{0, \ldots, m\}$.

Relate:

- Ron can either fulfil or skip order i, but if he can only fulfil it if he has at least a_i barrels in stock.
- If he fulfils it, his profit is $x(i, j) = p_i + x(i+1, j-a_i)$, if $j \geq a_i$.
- If he chooses to skip buyer i, his profit is $x(i, j) = x(i+1, j)$.
- Therefore the relation is $x(i) = \max\{p_i + x(i+1, j-a_i), x(i+1, j)\}$.

Topological order:

- $x(i, j)$ only depend on subproblems with strictly larger i, so acyclic.

Base:

- $x(n+1, j) = -sj$, since if there are no more buyers, there is a penalty of s cost per barrel.

Original:

- $x(1, m)$ is the maximum profit of m barrels to all n buyers.
- Store parent pointers to reconstruct which sales fulfill an optimal order.

Time:

- There are $(n+1)(m+1) = O(nm)$ subproblems.
- Each subproblem only requires a constant $O(1)$ amount of work.
- Therefore, the overall running time is $O(nm)$ which is pseudopolynomial in the size of the input.

Problem 8-2.

Subproblems:

- Let $x(i, j, c)$ be defined as the maximum possible score playing a game using a substring of pins from $i, \ldots, j$, and all the pins within this range must be knocked down if $c = 1$, but unconstrained if $c = 0$.
- For all $i \in \{0, \ldots, n\}$, $j \in \{i, \ldots, n\}$, and $c \in \{0, 1\}$.

Relate:

- Player can either choose to knock down pin i or not.
- If $c = 0$, the player can choose not to knock down pin i.
- Otherwise if $c = 1$, the player is forced to knock down pin i in three ways:
  - Knock down pin i by itself for v_i points.
  - Knock down pin i and $i + 1$ for $v_i \cdot v_{i+1}$ points.
  - Knock down with some pin k for $v_i \cdot v_k$ points, where $k \in \{i + 2, \ldots, j - 1\}$.
- $x(i, j, c) = \max \left\{ \begin{array}{ll} x(i+1, j, c), & \text{if } c = 0 \\ v_i + x(i+1, j, c), & \text{if } i < j \\ \max_{k \in \{i+1, \ldots, j-1\}} v_i \cdot v_k + x(i+1, k-1, 1) + x(k+1, j, c), & \text{if } i+1 < j \end{array} \right\}$

Topological order:

- $x(i, j)$ only depend on subproblems with strictly smaller $j - i$, so acyclic.

Base:

- In the case where there are no more pins, there is no more value to be earned, so $x(i, i, k) = 0$.
- For $i \in \{0, \ldots, n\}$ and $k \in \{0, 1\}$.

Original:

- The maximum score possible playing with all pins, and unconstrained: $x(0, n, 0)$.

Time:

- There are $O(n^2)$ subproblems, each with $O(n)$ work.
- Therefore, the overall running time is $O(n^3)$, which is polynomial in the size of the input.

Problem 8-3.

Subproblems:

- We define $x(k, s_1, s_2, s_3)$ as `True` if it is possible to partition suffix of items `A[k:]` into four subsets A_1, A_2, A_3, A_4, where $s_j = \sum_{a_i \in A_j} a_i$ for all $j \in \{1, 2, 3\}$, and `False` otherwise.
- For $k \in \{0, \ldots, n\}$ and $s_1, s_2, s_3 \in \{0, \ldots, m\}$.

Relate:

- We can place integer a_k into any of the four partitions.
- $x(k, s_1, s_2, s_3) = \text{OR} \left\{ \begin{array}{lr} x(k+1, s_1 - a_k, s_2, s_3), & \text{if } a_k \leq s_1 \\ x(k+1, s_1, s_2 - a_k, s_3), & \text{if } a_k \leq s_2 \\ x(k+1, s_1, s_2, s_3 - a_k), & \text{if } a_k \leq s_3 \\ x(k+1, s_1, s_2, s_3), & \text{always} \end{array} \right\}$

Topological order:

- $x(k, s_1, s_2, s_3)$ only depend on subproblems with strictly larger k, so acyclic.

Base:

- $x(n, 0, 0, 0) = \texttt{True}$, meaning we can partition zero integers into zero sum subsets.
- $x(n, s_1, s_2, s_3) = \texttt{False}$, for any $s_1, s_2, s_3 > 0$, meaning we cannot partition zero integers into any subset with positive sum.
- For $i \in \{0, \ldots, n\}$ and $k \in \{0, 1\}$.

Original:

- Let $m(0, s_1, s_2, s_3) = \left\{ \begin{array}{lr} \max\{s_1, s_2, s_3, m - s_1 - s_2 - s_3\}, & \text{if } x(0, s_1, s_2, s_3) = \texttt{True} \\ \infty, & \text{otherwise} \end{array} \right\}$
- Solution to the original problem: $\min\{m(0, s_1, s_2, s_3) : s_1, s_2, s_3 \in \{0, \ldots, m\}\}$.
- Store parent pointers to reconstruct each subset.

Time:

- There are $O(m^3 n)$ subproblems in total.
- Each subproblem requires $O(1)$ work.
- We need $O(n^3)$ time to compute the original.
- Therefore, overall $O(m^3 n)$ running time, which is pseudopolynomial in the size of the input.

Problem 8-4. First, scan through each of the $O(m^2n)$ at most length-m uncorrupted logs, and insert each word w into a hash table H mapping to frequency $f(w)$. This process computes all $f(w)$ for words appearing in any log in L_u directly in expected $O(m^3n)$ time, since the time to hash each word is linear in its length.

Second, we compute a restoration for each log l_i in L_c via dynammic programming in expected $O(m^3)$ time, leading to an expected $O(m^3n)$ running time in total, which is polynomial in the size of the input (there are at least $\Omega(nm)$ characters in the input).

Subproblems:

- Let $x(j)$ be defined as the maximum $\sum_{w \in R_{i,j}} f(w)$ for any restoration $R_{i,j}$ of suffix $l_i[j:]$.
- For $j \in \{0, \ldots, |l_i| \leq m\}$.

Relate:

- Guess the first word in $l_i[j:]$ and recur on the remainder.
- $x(j) = \max\{f(l_i[j:k]) + x(k) : k \in \{j+1, \ldots, |l_i|\}\}$.
- Where $f(w) = H(w)$ if $w \in H$, and 0 otherwise.

Toplogical order:

- $x(j)$ depends only on strictly larger j, so acyclic.

Base:

- $x(|l_i|) = 0$, if we have no log left to restore.

Original:

- Solution to the original problem is given by $x(0)$.
- Store parent pointers to reconstruct a restoration achieving value $x(0)$.

Time:

- There are $O(m)$ subproblems and $O(m^2)$ work for each subproblem, since we have $O(m)$ choices and each hash table lookup costs expected $O(m)$ time).
- Expected $O(m^3)$ running time per restoration.