Zig-Zag subsequence

Input: $x_1\ x_2\ \ldots\ x_n$

A subsequence $x_{i_1}, \ldots, x_{i_k}$ is a Zig-Zag subsequence if:

If $x_{i_a} \geq x_{i_{a+1}}$, then $x_{i_{a+1}} \leq x_{i_{a+2}}$

and if

$x_{i_a} \leq x_{i_{a+1}}$, then $x_{i_{a+1}} \geq x_{i_{a+2}}$

Example: 1 5 8 6 4 3 7 9

⇒ 1 8 3 9 is Zig-Zag

⇒ 8 6 7 is Zig-Zag

⇒ 1 5 6 4 is not.

Task: Find length of longest zig-zag subsequence

$D[i]$ – length of longest zig-zag subsequence that ends at i position that in the last pair decreased.

$I[i]$ – length of longest zig-zag subsequence that ends at i position that in the last pair increased.

$$D[i] = \max_{\substack{0 \leq j < i \\ x[j] \geq x[i]}} \{ I[j] \} + 1$$

$$I[i] = \max_{0 \leq j < i} \{ D[j] \} + 1.$$

Base case: $D[0] = I[0] = 1$

Running time: $O(n)$

Weighted Interval Scheduling

Input: n requests, with each request specifying a start time s_i, a finish time f_i and a value v_i.

Two requests are compatible if they do not overlap.

Goal: Find $S \subseteq \{1, \ldots, n\}$ of mutually compatible intervals that maximizes the sum of the values of the intervals in S: $\sum_{i \in S} v_i$.

⇒ Sort intervals by finish time. Consider the n-th interval.

⇒ The optimal solution may or may not include it.

⇒ $I(k)$ = max value using intervals 1 to k.

$$I(k) = \max\{I(k-1),\ I(p(k)) + v_k\}$$

$p(k) :=$ largest index i such that intervals i and j are disjoint.

2D-apples

⇒ Given a table A with $n \times m$ cells with each cell containing a # of apples.

⇒ At each step you can go down or right one cells

⇒ When you get to a cell, you get all apples in the cell

Task: Find max number of apples you can collect.

$S[i][j]$ = max number of apple you can collect in a path to cell i,j.

$$S[i][j] = \max\{S[i-1][j] \text{ if } i>0,\ S[i][j-1] \text{ if } j>0\} + A[i,j]$$

$$S[i][0] = \sum_{k=0}^{i-1} S[k][0]$$

$$S[0][j] = \sum_{k=0}^{j-1} S[0][k].$$

Running Time: $O(n \cdot m)$