ENS Lyon Camp. Day 3. Basic group.
Dynamic programming.
28 October

Contents

1 Dynamic Programming.

1.1 The Longest Common Subsequence (LCS).

Given two subsequense x and y the goal is to find the longest subsequence common to both of it. Let $lcs[i][j]$ be the lengths of LCS of prefixes of given subsequence that end in i and j, respectively.

$$lcs[i][j] = \begin{cases} 0, & \text{if } i = 0 \text{ or } j = 0 \\ lcs[i-1][j-1] + 1, & \text{if } x[i] = y[j] \\ \max\left(lcs[i][j-1], lcs[i-1][j]\right), & \text{if } x[i] \neq y[j] \end{cases} \tag{1}$$

The length of LCS of given sequences is stored in $lcs[n][m]$, where n and m — lengthes of given sequences. To restore answer, keep additional array $ans[i][j]$ — pair of indices shows where the optimal value come from.

1.2 The Longest Increasing Subsequence (LIS)

Given sequence a the goal is to find the longest increasing subsequense. Let $d[i]$ — the length of LIS of prefix that ends in i, so $d[i] = 1 + \max\limits_{j<i} d[j]$. Again to restore answer, keep additional array $ans[i]$ — index shows where the optimal value come from.

1.3 Knapsnack Problem.

Given a set of n items, each with a weight w_i and a cost c_i, determine the subset of items to put in a knapsnack so that the total weight is less than or equal to a given limit W and the total cost is as large as possible.

Let $dp[k][s]$ be maximal cost of items if the knapsnack limit is s and it it possible to use only first k items. It is obvious that: $dp[0][s] = 0$ for every s, and $dp[k][0] = 0$ for every k.

2 DP on segments.

3 DP on trees.

4 DP on masks.