CS660: Algorithms - Lecture 4

Instructor: Hoa Vu

San Diego State University

Longest Increasing Subsequence

- Given a list of numbers $A[1 \ldots n]$. Find the longest increasing subsequence.
- We define $LIS[i]$ as the length of the longest increasing subsequence that ends at $A[i]$.
- Let $A[0] = -\infty$ and $LIS[0] = 0$.
- Observe that the length of the longest increasing subsequence ending at $A[i]$ is equal to the length of the longest increasing subsequence ending at $A[j]$ for some $j < i$ where $A[j] \leq A[i]$ plus 1.
- Hence, $LIS[i] = 1 + \max_{0 \leq j < i : A[j] \leq A[i]} LIS[j]$. (how to turn this into code?)
- How to return the actual subsequence?
- Running time? $O(n^2)$.

Dynamic programming

- Define the dynamic programming table.
- Initialize the table (boundary cases).
- The filling rule.
- The order that you fill the table.

Subset sum

- Given a list of non-negative integers $A[1 \ldots n]$ and a target integer T.
- Decide if there is a subset of the numbers that sum to T. Return the subset if there exists one. Otherwise, output FALSE. Find the longest increasing subsequence.
- Let $SS[i, K] = true$ if some subsets of $A[1 \ldots i]$ sums to K.
- Initialize the table:
  - $SS[i, t] = TRUE$ if $t = 0$ (empty subset).
  - $SS[i, t] = FALSE$ if $t < 0$ or $i > n$.
- Then $SS[i, K] = true$ if one of the following two cases happens:
  a) $A[i]$ is in the subset that sums to K or
  b) $A[i]$ is not in the subset that sums to K.
- So we have

$$SS[i] = true \text{ iff } SS[i-1, K] = true \text{ or } SS[i-1, K - A[i]] = true.$$

- How to turn the above into actual code?
- Running time? $O(nT)$.

Edit distance

- Given two strings A, B (e.g., two DNA sequences).
- The minimum number of character insertions, deletions, and substitutions to transform A to B.
- Example: *FOOD* and *MONEY*. Edit distance is 4.
- Define the table ED where $ED[i, j]$ is the edit distance between $A[1 \ldots i]$ and $B[1 \ldots j]$.
- Initialization:
  - $ED[0, j] = j$ and $ED[j, 0] = 0$. Transforming the empty string to a string of length j requires j insertions.

Edit distance

- Initialization:
  - $ED[0, j] = j$ and $ED[j, 0] = 0$. Transforming the empty string to a string of length j requires j insertions.
- Filling the table:
- For $i = 1$ to m:
  - $ED[i, 0] \leftarrow i$.
  - For $j = 1$ to n:
    - $ins \leftarrow ED[i, j-1] + 1$
    - $del \leftarrow ED[i-1, j] + 1$
    - If $A[i] = B[j]$ then $rep \leftarrow ED[i-1, j-1]$
    - Else, $rep \leftarrow ED[i-1, j-1] + 1$.
    - $ED[i, j] \leftarrow min\{ins, del, rep\}$.
- Read 3.7.