Algorithm Design & Analysis (CSE222)

Lecture-11

Announcement

Raise assignment-1 concerns via google form in gc by today midnight.

Recap

- Longest Common Subsequence
- Maximum Job
- Edit Distance

Outline

- Edit Distance

- Weighted Subset

Levenshtein (Edit) Distance

Given two strings, edit distance measures the number of operations needed to transform one string into another.

Operations

- Insert
- Replace
- Delete

Input: ‘horse’ and ‘rose’
Replacing ‘h’ with ‘r’ results to ‘rorse’
Deleting ‘r’ results to ‘rose’.

Edit Distance

Input: A = horse and B = rose

Case-1: Do nothing
If A[5] equal to B[4]
then transform A[0-4] → B[0-3].

hors ros

Case-2: Insert
If A[3] not equal to B[3]
then transform A[0-3] → B[0-2].

A[0-3] → B[0-3]

hor → ro r

Case-3: Replace hor → ros
If A[3] not equal to B[3]
then transform A[0-2] → B[0-2].

ho → ro

Case-4: Delete hor → ros
If A[3] not equal to B[3]
then transform A[0-2] → B[0-3].

ho → ros

Edit Distance

<u>Subproblem:</u> EDist(A, B, i, j): Compute the edit distance to transform A[0-i]→B[0-j].

<u>Recurrence:</u>

$$\mathrm{EDist}(A,B,i,j) = \begin{cases} \mathrm{EDist}(A,B,i-1,j-1) & \text{if } A[i] == B[j] \\ 1 + \min \begin{cases} \mathrm{EDist}(A,B,i-1,j-1), \\ \mathrm{EDist}(A,B,i-1,j), \\ \mathrm{EDist}(A,B,i,j-1) \end{cases} & \text{else} \end{cases}$$

<u>Final Solution:</u> EDist(A, B, |A|, |B|)

Edit Distance

Initialize global matrix $ed = \{-1\}^{|A| \times |B|}$

Table filling or Memoization?

$\textsc{Edit-Distance}(A, B, i, j)$ $= ED(A, B, |A|, |B|)$

if $(A == \text{" "})$
    return j;
if $(B == \text{" "})$
    return i;
if $(ed(i, j) =! -1)$
    return $ed(i, j)$;
else
    if $(A[i] == B[j])$
        return $ed(i, j) = \textsc{Edit-Distance}(A, B, i-1, j-1)$;
    else
        insert $= \textsc{Edit-Distance}(A, B, i, j-1)$
        replace $= \textsc{Edit-Distance}(A, B, i-1, j-1)$
        delete $= \textsc{Edit-Distance}(A, B, i-1, j)$
        $ed[i, j] = 1 + \min\{\text{insert}, \text{replace}, \text{delete}\}$
        return $ed[i, j]$

Outline

- Edit Distance
- Weighted Subset

Weighted Subset

Let there are n items {1, 2, ..., n} and a weight function w: [n] → $R_{>0}$. Let W > 0.

Goal: Find subset S ⊆ {1, 2, ..., n} such that $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} w_i$ is maximum.

Trial-1:
Let n = 3 with weights W/2, W/2+1, W/2.
Ensure the maximum possible weighted item in the subset.
Sort the item by weights (say W/2+1, W/2, W/2).

Trial-2:
Sort the item by weights (say W/2, W/2, W/2+1) (It worked!).

Let n = 3 with weights 1, W/2, W/2?

$\{ \overset{1}{w/2+1}, \overset{2}{w/2}, \overset{3}{w/2} \}$

Weighted Subset

Subproblem: Wt(i): Optimum weight from first i elements.

Recurrence (handle cases): At iteration n

wt(n-1)

If n not in optimum then Wt(n) = Wt(n-1)

Else

Wt(n) = w_n + wt(n-1) ?

Weighted Subset

Subproblem: Wt(i,T): Optimum weight from first i elements with weight limited to T.

Recurrence (handle cases): At iteration n
If n not in optimum then Wt(n, W) = Wt(n-1, W)
Else
Wt(n,T) = w_n + Wt(n-1, W - w_n).

$$Wt(i, w) = \max\{Wt(i-1, w), Wt(i-1, w - w_i) + w_i\}$$

Final Solution: Wt(n, W)

Algorithm

$\textsc{Subset-Sum}(W, \{w_1, w_2, \ldots, w_n\})$

for $i = 0 \cdots n$
    $Wt[i, 0] = 0$
for $i = 0 \cdots W$
    $Wt[0, i] = 0$
for $i = 1 \cdots n$
    for $j = 1 \cdots W$
        if $w_i < j$
            $Wt[i, j] = \max\{Wt(n-1, w), Wt(n-1, w - w_n) + w_n\}$
        else
            $Wt[i, j] = Wt[i-1, j]$
return $Wt[n, W]$

Running time: O(nW)

Memoization?

Problem

Let there are n items {1, 2, ..., n}, a weight function w: [n] → $R_{>0}$ and a value function v: [n] → $R_{>0}$. Let W > 0.

Goal: Find subset S ⊆ {1, 2, ..., n} such that $\sum_{i \in S} w_i \leq W$ and the value of the subset if maximum.

Solution

Define the following

- Subproblem
- Recurrence
- Final Solution
- Algorithm and its running time.

Reference

Slides

Algorithm design by Kleinberg & Tardos - Chp-6.4