Longest Increasing Subsequence (LIS), Longest Common Substring, Longest Common Subsequence (LCS)

Pedro O. Pérez M., PhD.

Análisis y diseño de algoritmos avanzados
Tecnológico de Monterrey

pperezm@tec.mx

01-2023

Contenido

¿Qué es una subsecuencia?

Considera una cadena A = [a, b, c, d]. Una subsecuencia(también llamada subcadena) se obtiene eliminando 0 o más símbolos (no necesariamente consecutivos) de A.
Por ejemplo:

- Los ejemplos [a, b, c, d] , [a], [b], [c], [d], [a, d], [a, c], [b, d] son subsecuencias de A.
- Pero [d, b], [d, a], [d, a, c] no son subsecuencias de A.

Subsecuencia creciente más larga

Dado un arreglo de n símbolos comparables, dar la subsecuencia creciente más larga. Por ejemplo, A = [-7, 10, 9, 2, 3, 8, 8, 1], la subsecuencia creciente más larga es [-7, 2, 3, 8].

Procedure 1 LIS

```
Input: A : Array
    Aux : Array
    Aux[1] ← 1
    for i ← 2 to n do
        count ← 0
        for j ← 1 to (i − 1) do
            if A[j] < A[i] then
                count ← MAX(count, Aux[j])
            end if
        end for
        Aux[i] ← count + 1
    end for
    count ← 0
    for i ← 1 to n do
        count ← MAX(count, Aux[i])
    end for
    return count
```

Subsecuencia común más larga

Dado dos cadenas, A y B, determinar la subsecuencia común más larga de ambas cadenas. Por ejemplo, si A = [c, d, c, c, f, g, e] y B = [e, c, c, e, g, f, e], ¿cuál la subsecuencia común más larga?

Ahora veamos como se plantea el problema: considere dos secuencias A = [a_1, a_2, ..., a_m] y B = [b_1, b_2, ..., b_n] . Para resolver el problema nos fijaremos en los últimos dos símbolos: a_m y b_n . Como podemos ver hay dos posibilidades:

- Caso 1: $a_m = b_n$. En este caso, la subsecuencia común más larga debe contener a_m . Simplemente basta encontrar la subsecuencia común más larga de [a_1, a_2, ..., a_{m-1}] y [b_1, b_2, ..., b_{n-1}].
- Caso 2: $a_m \neq b_n$. En este caso, puede hacerse corresponder [a_1, a_2, ..., a_m] con [b_1, b_2, ..., b_{n-1}] y también [a_1, a_2, ..., a_{m-1}] con [b_1, b_2, ..., b_n], y nos quedamos con el mayor de los dos resultados.

Procedure 2 LCS

Input: $A, B : Array$
  $M : Matrix[0..A.length][0..B.length]$
  $m \leftarrow A.length$
  $n \leftarrow B.length$
  $INIT(M, 0)$
  for $i \leftarrow 1$ to m **do**
    for $j \leftarrow 1$ to n **do**
      if $A[i] = B[j]$ **then**
        $M[i][j] \leftarrow 1 + M[i-1][j-1]$
      else
        $M[i][j] \leftarrow MAX(M[i-1][j], M[i][j-1])$
      end if
    end for
  end for
  return $M[m][n]$

Subcadena común más larga

El problema de la subcadena común más larga (Longest Common Substring) consiste en encontrar aquella subcadena continua que puede coincidir en todos las cadenas de entrada no necesariamente del mismo tamaño. Este problema puede arrojar múltiples soluciones. Por ejemplo, teniendo los strings: “AABCAB” y “CABCBABACC”, el tamaño de la subcadena más larga es 3: “ABC”, “CAB” y “ABA”.

Ahora veamos como se plantea el problema: considere dos secuencias A = [a_1, a_2, ..., a_m] y B = [b_1, b_2, ..., b_n] . Para resolver el problema nos fijaremos en los últimos dos símbolos: a_m y b_n . Como podemos ver hay dos posibilidades:

- Caso 1: $a_m = b_n$. En este caso, la subcadena común más larga debe contener a_m . Primero debemos encontrar la subcadena común más larga de [a_1, a_2, ..., a_{m-1}] y [b_1, b_2, ..., b_{n-1}] + 1, y luego comparar ese resultado con el máximo previo.
- Caso 2: $a_m \neq b_n$. En este caso, la longitud más larga será cero.

Procedure 3 LCS

```
Input: A, B : Array
  M : Matrix[0..A.length][0..B.length]
  m ← A.length
  n ← B.length
  for i ← 0 to m do
    for j ← 0 to n do
      if i = 0 or j = 0 then
        M[i][j] ← 0
      else
        if A[i] = B[j] then
          M[i][j] ← 1 + M[i − 1][j − 1]
          maximum ← MAX(M[i][j], max)
        else
          M[i][j] ← 0
        end if
      end if
    end for
  end for
  return maximum
```