Subsecuencia creciente más larga (Longest Increasing Subsequence a.k.a. LIS)

Nicolás Álvarez

Departamento de Ciencias e Ingeniería de la Computación. Primer cuatrimestre 2021

La subsecuencia creciente más larga (LIS por sus siglas en inglés) es una subsecuencia dentro de un arreglo de números con un orden creciente.

La subsecuencia creciente más larga (LIS por sus siglas en inglés) es una subsecuencia dentro de un arreglo de números con un orden creciente.

Los números dentro de la subsecuencia tienen que ser únicos y estar ordenados de forma ascendente.

La subsecuencia creciente más larga (LIS por sus siglas en inglés) es una subsecuencia dentro de un arreglo de números con un orden creciente.

Los números dentro de la subsecuencia tienen que ser únicos y estar ordenados de forma ascendente.

Es importante tener en cuenta que los elementos de la secuencia no necesariamente tienen que estar en lugares consecutivos dentro del array.

La subsecuencia creciente más larga (LIS por sus siglas en inglés) es una subsecuencia dentro de un arreglo de números con un orden creciente.

Los números dentro de la subsecuencia tienen que ser únicos y estar ordenados de forma ascendente.

Es importante tener en cuenta que los elementos de la secuencia no necesariamente tienen que estar en lugares consecutivos dentro del array.

¿Podemos diseñar un algoritmo eficiente que encuentre la longitud de la subsecuencia creciente más larga?

Ejemplo

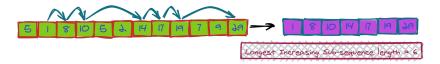


Figure 1: Ejemplo de LIS

▶ entrada: {1,5,2,7,3}

• entrada: $\{1,5,2,7,3\}$ LIS = 3

```
► entrada: {1,5,2,7,3}

LIS = 3

{1,5,7},{1,2,3},{1,2,7}
```

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- ► entrada: {13,1,3,4,8,4}

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- entrada: {13,1,3,4,8,4}
 LIS = 4

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- entrada: {13,1,3,4,8,4}
 LIS = 4
 {1,3,4,8}

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- entrada: {13,1,3,4,8,4}
 LIS = 4
 - $\{1,3,4,8\}$
- entrada: {13,1,3,4,8,19,17,8,0,20,14}

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- ► entrada: {13,1,3,4,8,4} LIS = 4 {1,3,4,8}
- ► entrada: {13,1,3,4,8,19,17,8,0,20,14} LIS = 6

- ► entrada: {1,5,2,7,3} LIS = 3 {1,5,7},{1,2,3},{1,2,7}
- entrada: {13,1,3,4,8,4}
 LIS = 4
 {1,3,4,8}
- ► entrada: {13,1,3,4,8,19,17,8,0,20,14} LIS = 6 {1,3,4,8,17,20},{1,3,4,8,19,20}

Planteo como Programación Dinámica

LIS(i) = "Longitud de la subsecuencia creciente más larga que termina en la posicion i"

Planteo como Programación Dinámica

LIS(i) = "Longitud de la subsecuencia creciente más larga que termina en la posicion i"

Ejecución

secuencia	13	1	3	4	8	19	17	8	0	20	14
LIS											

Figure 2: Tabla vacía

Planteo como Programación Dinámica

LIS(i) = "Longitud de la subsecuencia creciente más larga que termina en la posicion i"

Ejecución

secuencia	13	1	3	4	8	19	17	8	0	20	14
LIS											

Figure 2: Tabla vacía

secuencia	13	1	3	4	8	19	17	8	0	20	14
LIS	1	1	2	3	4	5	5	4	1	6	5

Figure 3: Tabla llena

Formula

Formula

$$LIS(i) = \begin{cases} 1 \text{ si } \nexists j < i \text{ tal que } secuencia[j] < secuencia[i] \\ 1 + \max_{j < i \land secuencia[j] < secuencia[i]} LIS[j] \end{cases}$$

La implementación es simple y queda como ejercicio.

Comentarios finales

Existen soluciones más eficientes que tardan un tiempo de $O(n \log n)$. Pueden buscar *Patience sorting* y su relación con el problema de LIS.

Comentarios finales

Existen soluciones más eficientes que tardan un tiempo de $O(n \log n)$. Pueden buscar *Patience sorting* y su relación con el problema de LIS.

Una aplicación interesante de este problema es generar un *diff* de archivos que sea más fácil de leer para un humano. Ver **comentario en stackoverflow**.