

# Tutorial: Maior subsequência Crescente

Arthur Andrade D'Oliveira

## 1 Solução do Problema

O problema de encontrar a maior subsequência crescente (*Longest Increasing Subsequence* - LIS) é um clássico que pode ser resolvido de forma eficiente por meio de **programação dinâmica**. A ideia central é focar no final da subsequência: para cada elemento da sequência original, calculamos qual é a maior subsequência crescente que termina exatamente nele.

### 1.1 Definição do Subproblema

Seja  $a$  o nosso vetor de números inteiros com  $N$  elementos, indexado de 0 a  $N - 1$ . Definimos o nosso estado da programação dinâmica como:

$d[i]$  = o comprimento da maior subsequência crescente que termina obrigatoriamente no índice  $i$ .

A resposta final para o tamanho da maior subsequência não será necessariamente  $d[N - 1]$ , mas sim o maior valor alcançado em todo o vetor  $d$ , ou seja,  $\max(d[i])$  para  $0 \leq i < N$ .

### 1.2 Função de Transição

Para determinar o valor de  $d[i]$ , precisamos olhar para todos os elementos anteriores à posição  $i$ , ou seja, um índice  $j$  tal que  $0 \leq j < i$ .

Se encontrarmos um elemento anterior que seja estritamente menor que o elemento atual ( $a[j] < a[i]$ ), isso significa que podemos estender a subsequência que terminava em  $j$  anexando o elemento  $a[i]$  ao final dela. O novo comprimento seria  $d[j] + 1$ . Avaliamos todos os possíveis índices  $j$  válidos e mantemos o que gera o maior comprimento:

$$d[i] = \max_{\substack{0 \leq j < i \\ a[j] < a[i]}} (d[j] + 1)$$

### 1.3 Casos Base

O caso base é bastante intuitivo: na pior das hipóteses, qualquer elemento isolado forma, por si só, uma subsequência válida de comprimento 1. Portanto, a nossa inicialização padrão para todos os estados é:

$$d[i] = 1 \quad \text{para todo } 0 \leq i < N$$

### 1.4 Recuperação da Decomposição (Elementos da subsequência)

Como o problema exige a impressão dos elementos que formam a subsequência, precisamos reconstruir o caminho ótimo. Para isso, utilizamos um vetor auxiliar  $p[i]$  (de *parent* ou predecessor), inicializado com  $-1$ .

Durante a transição, sempre que atualizamos  $d[i]$  com um valor maior proveniente de um  $d[j] + 1$ , registramos o índice de origem definindo  $p[i] = j$ .

Após calcularmos todo o vetor  $d$ , localizamos o índice **pos** que contém o valor máximo absoluto de  $d$ . A partir de **pos**, recuperamos os elementos da subsequência percorrendo o caminho reverso: adicionamos  $a[\mathbf{pos}]$  à nossa lista de resposta e saltamos para o predecessor atualizando  $\mathbf{pos} = p[\mathbf{pos}]$ . Repetimos o processo até que **pos** seja  $-1$ . Como resgatamos os elementos do final para o começo, basta inverter a lista resultante para apresentar a subsequência na ordem crescente original.