Is Bigger Smarter?

Johanna Beltran y Diego Triviño

2012

Índice

1. Introducción

Is Bigger Smarter? es un problema de programación dinámina, el cual encontramos en el juez virtual UVA con el número **10131**. Este documento busca mostrar una de las tantas soluciones desde el enfoque matemático, el objetivo es realizar el código con la solución del problema en cualquier lenguaje de programación con la ayuda de este documento.

2. Definición del problema

Se deben tomar los datos de una colección de elefantes y mostrar el subconjunto mas grande de manera que los pesos están aumentando, pero el coeficiente intelectual disminuyendo.
Se deben tener en cuenta las siguientes restricciones para la solución del problema:

1. Cada elefante que se ingresa debe tener sus respectivos datos, es decir peso y coeficiente intelectual.
2. Tanto el peso como el coeficiente intelectual se manejaran en un rango entre 1 y 10000.
3. Solo se podran ingresar hasta 1000 elefantes.
4. Dos elefantes pueden tener el mismo peso ó el mismo coeficiente intelectual, incluso el mismo peso y coeficiente intelectual.
5. Todas las desigualdades son estrictas en la solución: Los pesos deben ser estrictamente creciente, y el coeficiente intelectual debe ser estrictamente decreciente.

2.1. Entrada

Se ingresa una pareja de números enteros; el primero representa el peso en kilogramos y el segundo representa el coeficiente intelectual del elefante.

EJEMPLO

2 8

1 9

3 7

2.2. Salida

Se imprime la longitud de la secuencia mas grande de elefantes seguido por los elefantes que la conforman.

EJEMPLO ANTERIOR

3

2

1

3

3. Modelamiento matemático

Un elefante se modela como una pareja ordenada (P, IQ) donde P:peso, IQ:coeficiente intelectual y :

$$1 \leq P \leq 10000 \wedge 1 \leq IQ \leq 10000$$

Dada una lista de elefantes de la forma:

$e_i = (P, IQ)$ donde $1 \leq i \leq 1000$

Una entrada esta dada por una sucesión de elefantes e_1,e_2,e_3, ... , e_n.

4. Planteamiento de la solución

Se va a ordenar la lista de elefantes ingresada, de manera que el coeficiente intelectual sea descendente; siendo asi se puede encontrar el elefante estrictamente anterior que cumpla con las restricciones, es decir mayor coeficiente intelectual y menor peso, llevandonos a encontrar una pequeña subsecuencia junto con su tamaño, donde el final de esta es el elefante que estamos evaluando. Al encontrar que el elefante anterior ya tiene una subsecuencia formada se suma al tamaño de la subsecuencia que se esta evaluando. Es importante saber cual es el tamaño y cuales son los elefantes que conforman la secuencia mas larga, por lo tanto utilizamos una matriz en la cual, en la primera fila se encontraran los tamaños de las subsecuencias de cada elefante ingresado y las demás filas seran los elefantes anteriores al evaluado.

Se genera un vector B de elefantes ordenados por IQ de manera descendente.

$$\vec{B} = \langle f1, f2, f3, \ldots fn \rangle$$

f es un elefante donde:
$f_i = (P_i, IQ_i), f_{i-1} = (P_{i-1}, IQ_{i-1})$ tal que $IQ_i < IQ_{i-1}$
Principalmente se piensa en guardar los tamaños de las subsecuencias formadas con cada elefante en un vector T:

$$\vec{T} = \langle t1, t2, t3, \ldots tn \rangle$$

donde la ecuación recurrente del mayor tamaño de secuencia de elefantes es:
$t_i = 1+$ MAX $\{t\,(j) : 1 \leq j < i \wedge f_j$.Peso $< f_i$.Peso $\}$
Para guardar no solo los tamaños sino también los elefantes que conforman la secuencia se crea una matriz L:

$$L = \begin{pmatrix} \mathrm{t}_{00} & \mathrm{t}_{01} & \mathrm{t}_{02} \ldots t_{0n} \\ \mathrm{g}_{10} & \mathrm{g}_{11} & \mathrm{g}_{12} \cdots g_{1n} \\ \mathrm{g}_{20} & \mathrm{g}_{21} & \mathrm{g}_{22} \cdots g_{2n} \\ \mathrm{g}_{30} & \mathrm{g}_{31} & \mathrm{g}_{32} \cdots g_{3n} \\ . & . & . \\ . & . & . \\ . & . & . \\ \mathrm{g}_{n0} & \mathrm{g}_{n1} & \mathrm{g}_{n2} \cdots g_{nn} \end{pmatrix}$$

Donde cada t_{0i} es el tamaño de la subsecuencia más grande que termina en el elefante f_i, y g_{ij} son los elefantes que conforman la subsecuencia.
Al obtener la matriz podemos encontrar el tamaño más grande en la primera fila; siendo así, podemos obtener los elefantes de la subsecuencia en la misma columna del tamaño pero variando la fila hasta encontrar el último elefante.

5. Conclusiones

1. Para este problema es necesario y conveniente utilizar programación dinámica porque no existe la necesidad de recalcular aquellas soluciones que ya teníamos en algún momento de la ejecución; ahorrando tiempo y codificación en la solución.

2. Para un problema puede existir muchas soluciones; tenga en cuenta que el vector de elefantes inicial puede ser ordenado tanto por coeficiente intelectual como por peso, manteniendo siempre las restricciones de la solución.