Identificación:

Santiago Pardo - 202013024

Kevin Cohen - 202011864

Algoritmo de solución:

El problema A busca calcular la convolución ponderada de una empresa en un tiempo n. Este indicativo se calculaba con la siguiente formula.

$$cp(r,n) = (+k|0 \leq k \leq n : r(k) * r(n-k))$$

La función r de la expresión era una ecuación recursiva dada por las siguientes expresiones:

r(0) = A

r(1) = B

r(k+2) = C * r(k) + D*r(k+1)

Al ser una ecuación de recurrencia optamos por solucionar este método con programación dinámica, ya que de esta manera utilizando una estructura de datos podemos llegar de manera constante a las soluciones previas y de esta manera maximizar a eficiencia sacrificando espacio en memoria.

Para hallar la solución primero se hallaron todas las soluciones de la ecuación r hasta el valor n iterando sobre un arreglo que tuviera los casos bases y a partir de ellos obtener los siguientes resultados. Todas estas soluciones se almacenan en un arreglo de tamaño n. Después, se realiza la sumatoria de la ecuación cp, esta sumatoria itera n veces ya que es de cero hasta n, pero gracias a que se almacenaron todas las respuestas en un arreglo utilizando programación dinámica, se pueden obtener los resultados de manera constante y por consecuencia manteniendo la eficiencia de la sumatoria.

La precondición es la siguiente:

$$\{-1 \leq A \leq 1, -1 \leq B \leq 1, -1 \leq C \leq 1, -1 \leq D \leq 1, n > 0\}$$

La postcondición es la siguiente, donde s es lo que retorna el algoritmo:

$$\{s = (+k|0 \leq k \leq n : r(k) * r(n-k))\}$$

Existía la alternativa de no hacer programación dinámica, solamente utilizando la ecuación de recurrencia y calculándola cada vez que se necesitara, esta solución no fue tomada en cuenta ya que a pesar de que no hace gasto en memoria, el calcular de manera recursiva cada vez que se necesite un resultado de la ecuación, haría que el algoritmo se demorara demasiado al calcular la sumatoria

Análisis de Complejidades:

Con lo anteriormente mencionado podemos hacer un análisis de las complejidades espaciales y temporales. Para hallar la complejidad espacial revisamos las estructuras de datos usadas, en la solución presentada solo se utilizó un arreglo de n posiciones, donde se almacenaron todas las soluciones de la ecuación r hasta la variable n, por lo que la complejidad espacial es igual a n.

Por otro lado, para obtener la complejidad temporal se deben de analizar todos los recorridos y seleccionar el que sea mayor. Para esta solución se plantearon dos recorridos, el primero que busca obtener todas las soluciones de la ecuación r hasta la variable n y el segundo que busca realizar la sumatoria de 0 hasta n. El primer recorrido tiene una complejidad O(n) ya que en un solo recorrido se pueden obtener todas las soluciones de r, esto se logró almacenando todas las soluciones pasadas en un arreglo, de modo que cada vez que se necesitara se llegaba a esta de manera constante. Por otro lado, la complejidad de del segundo recorrido también es de orden O(n), ya que al hacer la sumatoria de 0 hasta n no toca hacer otro recorrido para hallar un valor de r, ha que todas las respuestas de este están almacenadas en un arreglo, por lo que también llegar a estas soluciones era de tiempo constante y no afectaban en tiempo de calcular la sumatoria de 0 hasta n.

Por lo tanto, al tener un recorrido separado que son de orden O(n) obtenemos que la complejidad temporal de esta solución es de orden O(n).

Comentarios finales

La solución propuesta en términos de complejidades espaciales y temporales es buena ya que no ocupa demasiado espacio en memoria, ni se demora en ejecutarse durante mucho tiempo, lo que hace que el algoritmo sea útil y viable en contextos de números grandes.