TRAIN

Sara Chica, Rodrigo Gualtero

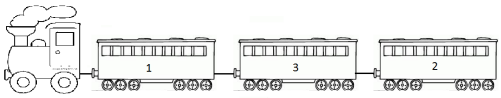
24 de Noviembre, 2012

Índice

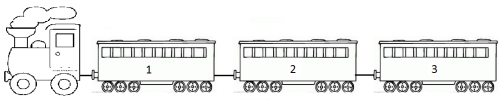
1. Introducción

Este es un problema de la UVA, identificado con el código *299*, en el cual se desea ordenar un vector de números en forma ascendente, donde el vector es un grupo de vagones el cual debe ser organizado de esta forma.
Sin embargo lo importante en este no es organizarlo, sino dar el menor número de pasos que se deben hacer para ordenar el tren.
A continuación se presentan 2 ejemplos de nodos:

En el primer ejemplo se presenta un tren de 3 vagones:

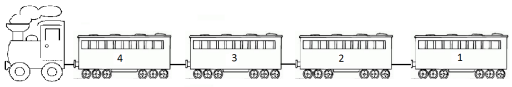


Ejemplo 1.1: Tren desordenado.

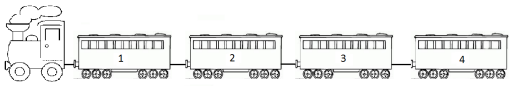


Ejemplo 1.2: Tren ordenado.

En el segundo ejemplo se presenta un tren de 4 vagones:



Ejemplo 2.1: Tren desordenado.



Ejemplo 2.2: Tren ordenado.

2. Definición del problema

Este problema busca ordenar una lista de trenes, cada uno marcado con un número, en forma ascendente, es decir de menor a mayor.

2.1. Entrada

En la primera linea entra un valor que determina el número casos que habrán.
Seguido a esto entran cada uno de los casos de prueba en dos partes; la primera consta de un número que indica la longitud del tren (Menor que 50), y finalmente entran todos los vagones en el orden actual.

2.2. Salida

Imprime la siguiente frase:

Optimal train swapping takes X swaps.

donde X es un entero que indica la cantidad de pasos que tuvo que hacer para organizar el vector.

3. Modelamiento matemático

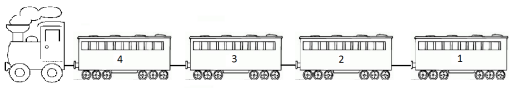
Se sabe que el tren está organizado si:

$$Ordenado \equiv (\forall i, j | 0 < i < j < 50 : V_{i+1} = V_j)$$

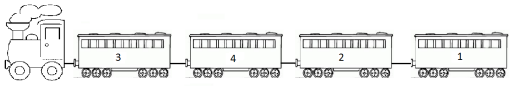
Sobre este problema se podría extender un árbol de posibilidades, el cual tendría todas las posibles permutaciones que habrían, donde una de estas es la que satisface la condición de salida: Tener todos los vagones ordenados.
Analizando el ejemplo 1, existen 6 posibles permutaciones de las cuales sólo serviría 1.

4. Planteamiento de la Solución

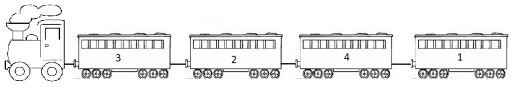
Aunque abrir un árbol de posibilidades permite analizar mejor el problema, no es la mejor solución, debido a que es poco eficiente; es por ello que existen varios algoritmos de ordenamiento, los cuales permiten organizar una lista en una secuencia dada. Para este problema se usa el algoritmo burbuja.
Ordenamiento por Burbuja: Consiste en organizar el vector revisando cada elemento de la lista y comparándolo con el siguiente, de esta forma si están en el orden equivocado se deben intercambiar.
Aplicando esto al ejemplo 2 se tendría lo siguiente:



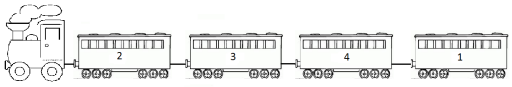
Ejemplo 2.3: Tren Original.



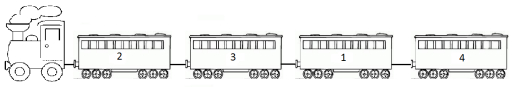
Ejemplo 2.4: Paso 1.



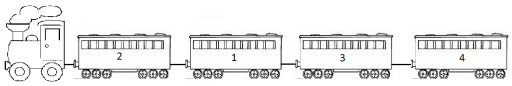
Ejemplo 2.5: Paso 2.



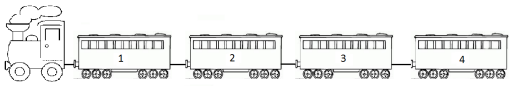
Ejemplo 2.6: Paso 3.



Ejemplo 2.7: Paso 4.



Ejemplo 2.8: Paso 5.



Ejemplo 2.9: Paso 6.

5. Conclusiones

1. Conocer los algoritmos de ordenamiento es importante para optimizar algunos otros algoritmos como los de búsqueda y fusión; ya que estos pueden requerir listas ordenadas para tener una ejecución rápida.

2. El método de ordenamiento burbuja es el algoritmo de ordenamiento más usado en los lenguajes de programación.