

PRÁCTICA 3

Algoritmos voraces

Álvaro Maximino Linares Herrera

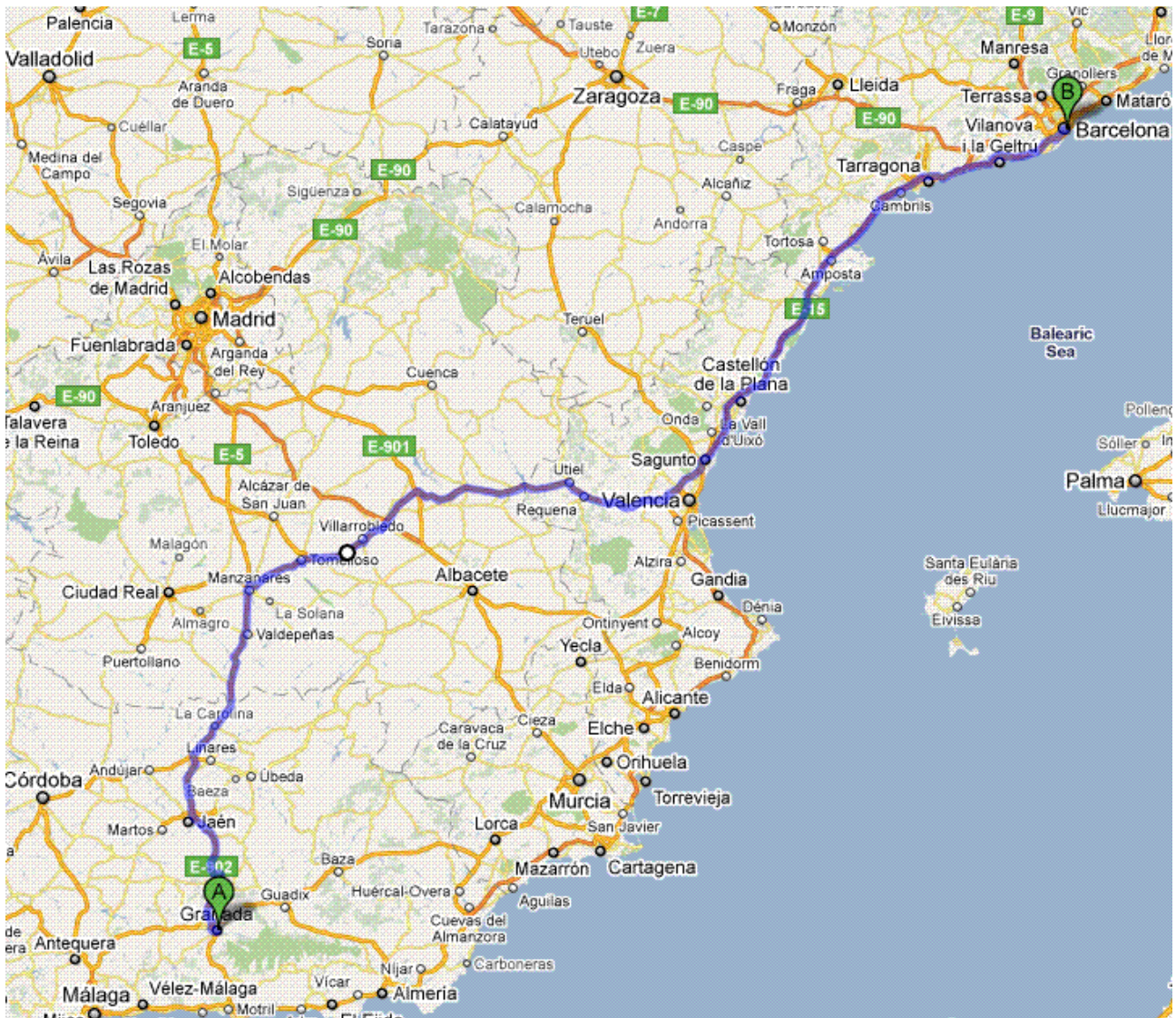


Nota:

El código del problema del camionero lo he implementado a partir del pseudocódigo del libro Técnicas de Diseños de Algoritmos, sus autores son Rosa Guerequeta García y Antonio Vallecillo Moreno. Este libro me ha ayudado a implementar algunos códigos y me ha ayudado a entender mejor la asignatura.

En el apartado ¿Encuentra siempre la solución óptima? Lo que he hecho a sido una pequeña reflexión de que siempre encontrará la solución óptima.

PROBLEMA DEL CAMIONERO:



El problema trata sobre que un camionero tiene que hacer una determinada ruta, Granada-Barcelona, y como tiene prisa debe de parar en el menor número posible de gasolineras del recorrido, su tanque de gasolina lleno le permite recorrer n kilómetros sin parar, luego para determinar la separación que hay entre las gasolineras lo haremos aleatoriamente de modo que en un vector llamado gasolineras de un determinado tamaño, correspondiente al número de gasolineras que queramos poner en el recorrido, haremos lo siguiente:

```
for(int h=0; h<tam; h++)  
    gasolineras[h]=rand()%n;
```

Siendo tam el tamaño del vector, gasolineras el vector en si y n el número de kilómetros que puede recorrer sin repostar. Al usar `rand()%n` nos aseguramos que nos genere números entre 0 y $n-1$ de modo que nunca se pasaría de los n kilómetros que podemos recorrer.

La idea del problema es construir un algoritmo voraz que nos diga en cuales de las “tam” gasolineras del camino nos debemos de parar de modo que sean las menores posibles ya que “vamos con prisa”.

Código:

El código de problema lo he implementado sobre un pseudocódigo de Internet y es el siguiente:

```
void Camionero(int n, int *gasolineras, int tam, bool *&sol){
    int que_gasolinera=-1;
    int numero_km=0;
    do{
        do{
            que_gasolinera++;
            numero_km+=gasolineras[que_gasolinera];
        }while( (numero_km<n) && (que_gasolinera<tam) );
        if(numero_km>n){
            que_gasolinera--;
            sol[que_gasolinera]=true;
            numero_km=0;
        }
    }while(que_gasolinera<tam);
}
```

Explicación del código:

A la función Camionero le paso cuatro parámetros los cuales son:

- Un entero n: El número de kilómetros que podemos recorrer sin pararnos a repostar.
- Un puntero de enteros gasolinera: Será usado como vector, en el que se incluirán las distancias entre las distintas gasolineras del camino.
- Un entero tam: El tamaño del vector, correspondiente al número de gasolineras del camino.
- Un puntero de booleanos sol: Corresponde con la decisión de si nos pararemos en una gasolinera o no, está pasado por referencia ya que durante la ejecución del algoritmo lo modificaremos, en el main lo inicializamos a false.

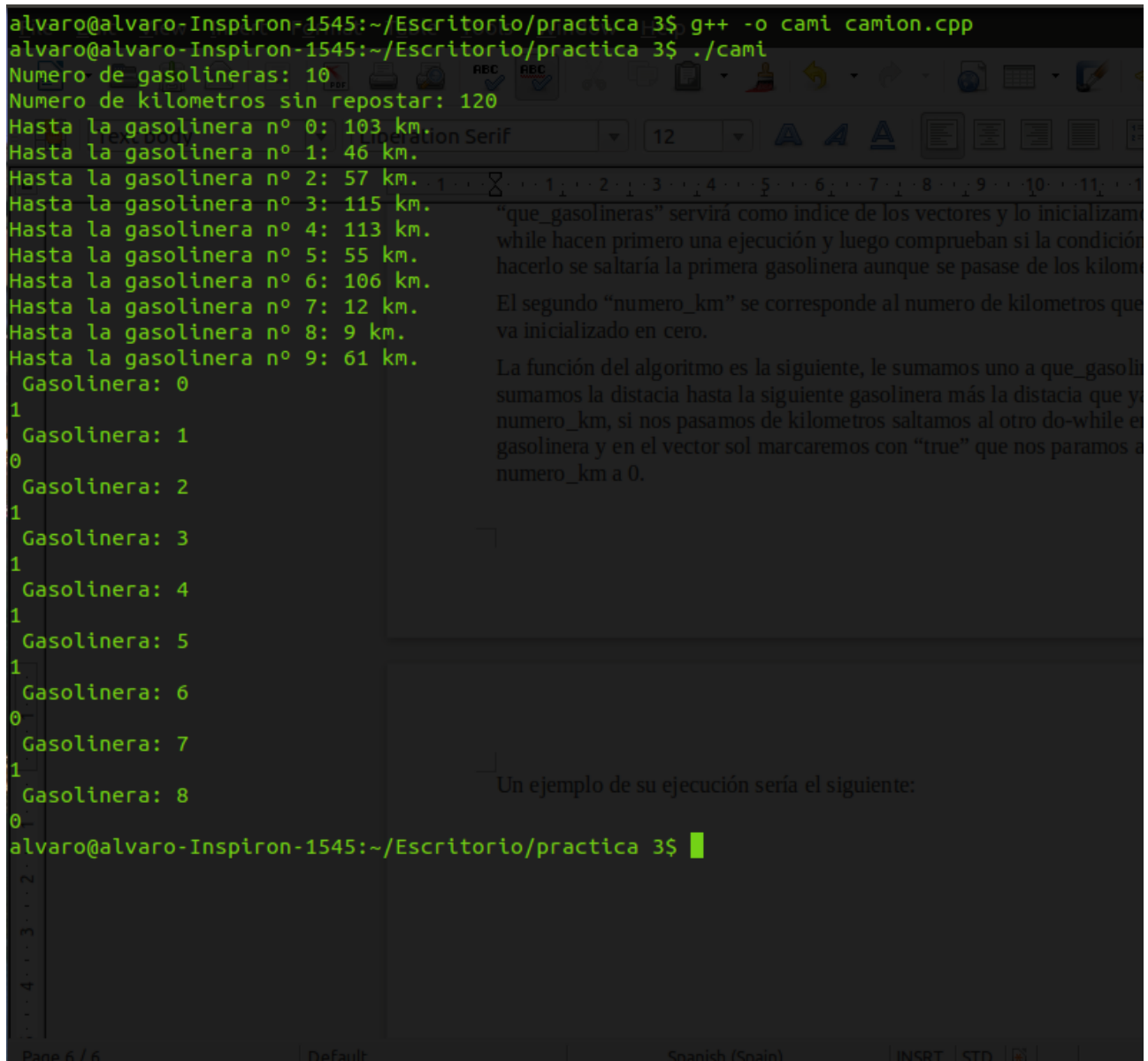
El código es bastante sencillo, como ve usted, declaramos al principio dos enteros, el primero “que_gasolineras” servirá como índice de los vectores y lo inicializamos en -1 porque los bucles do-while hacen primero una ejecución y luego comprueban si la condición es cierta o no y de no hacerlo se saltaría la primera gasolinera aunque se pasase de los kilómetros que podemos recorrer.

El segundo “numero_km” se corresponde al numero de kilómetros que llevamos recorridos, por eso

va inicializado en cero.

La función del algoritmo es la siguiente, le sumamos uno a `que_gasolineras` y a `numero_km` le sumamos la distancia hasta la siguiente gasolinera más la distancia que ya tuviésemos acumulada en `numero_km`, si nos pasamos de kilómetros saltamos al otro `do-while` en el que retrocederemos una gasolinera y en el vector `sol` marcaremos con “true” que nos paramos a repostar allí y pondremos el `numero_km` a 0.

Un ejemplo de su ejecución sería el siguiente:



```
alvaro@alvaro-Inspiron-1545:~/Escritorio/practica 3$ g++ -o cami camion.cpp
alvaro@alvaro-Inspiron-1545:~/Escritorio/practica 3$ ./cami
Numero de gasolineras: 10
Numero de kilometros sin repostar: 120
Hasta la gasolinera nº 0: 103 km.
Hasta la gasolinera nº 1: 46 km.
Hasta la gasolinera nº 2: 57 km.
Hasta la gasolinera nº 3: 115 km.
Hasta la gasolinera nº 4: 113 km.
Hasta la gasolinera nº 5: 55 km.
Hasta la gasolinera nº 6: 106 km.
Hasta la gasolinera nº 7: 12 km.
Hasta la gasolinera nº 8: 9 km.
Hasta la gasolinera nº 9: 61 km.
Gasolinera: 0
1
Gasolinera: 1
0
Gasolinera: 2
1
Gasolinera: 3
1
Gasolinera: 4
1
Gasolinera: 5
1
Gasolinera: 6
0
Gasolinera: 7
1
Gasolinera: 8
0
alvaro@alvaro-Inspiron-1545:~/Escritorio/practica 3$
```

Como se puede apreciar se ven el número de kilómetros que podemos recorrer, 120, el número de gasolineras que hay, 10, y muy claramente la distancia que hay de una a otra, y también vemos si paramos o no, en caso de pararnos hay un 1 indicando que es true esa gasolinera y en caso de no pararnos hay un 0 indicando que es false.

¿Encuentra siempre la solución óptima?

Obviamente siempre encuentra la solución óptima, pues no podría darse ningún caso en el que el camionero, usando otro conjunto solución de gasolineras, parase menos veces pues se pasaría de kilómetros y no tendría gasolina, quedándose tirado en una cuneta, cuando con el método aquí descrito llegaría sin problemas a Barcelona (o cualquier otro destino), pues con los dos bucles `do-while` tenemos la ventaja de que primero ejecutamos y si vemos que se pasa volvemos hacia atrás para que reposte y reiniciemos los kilómetros que le quedan al tanque de gasolina.