ANÁLISIS Y DISEÑO DE ALGORITMOS

Backtracking

Práctica 8 de laboratorio

Entrega: Hasta el domingo 6 de Mayo de 2018, 23:55h (a través de Moodle)

Asignación de coste mínimo III

Una comarca compuesta de n aldeas comunicadas mediante una red de carreteras que conforman un grafo conexo no necesariamente completo[1], está planificando la instalación de g gasolineras ($0 < g \leq n$) para dar servicio a todos sus vehículos. De cada vehículo se conoce la aldea en la que está censado y para repostar deberá ir a la población más cercana que disponga de estación de servicio. Se pretende conocer en qué aldeas habría que ubicar las gasolineras para minimizar el total de las distancias que han de recorrer todos los vehículos para ir a la gasolinera más cercana.

Se pide, aplicar el método ***Backtracking*** (vuelta atrás) para obtener la mejor ubicación de las gasolineras.

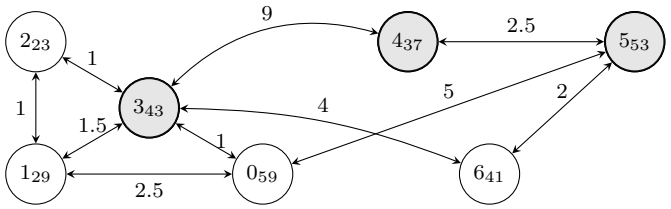

Figura 1: Posible ubicación de las tres gasolineras en los vértices sombreados en gris. Los habitantes de las aldeas 0, 1 y 2 irían a repostar a la aldea 3 (ya que es donde está la gasolinera mas cercana), y los de la aldeas 6 irían a la 5. Los de las demás aldeas no se moverían ya que dispondrían de gasolinera en su aldea. El coste asociado a esta solución (suma total de distancias a recorrer) viene dado por la expresión $59 \cdot 1 + 29 \cdot 1,5 + 23 \cdot 1 + 41 \cdot 2 = 207,5$. **Esta ubicación de las gasolineras es la mejor.**

Al igual que en la práctica anterior, las distancias se suministrarán al programa mediante una matriz cuadrada d, $n \times n$, donde, para simplificar, $d_{ii} = 0$ y $d_{ij} = d_{ji} \in R^+$ contiene la distancia del camino más corto entre la aldea i y la aldea j, $\forall i, j : 0 \leq i, j \leq n-1$ (de esta manera no es relevante conocer cuál es la carretera que produce el camino más corto o cuántas aldeas intermedias se atraviesan). Por otra parte, el número de vehículos censados en cada población vendrá dado en un vector v con $v_i \in N$.

- **Nombre del programa, opciones y sintaxis de la orden.**

  El programa a realizar se debe llamar **mca_bt**. La orden tendrá la siguiente sintaxis:

  mca_bt -f `fichero_entrada`

  El problema a resolver se suministrará codificado en un fichero de texto cuyo nombre se recogerá a través de la opción -f. Su formato y contenido será:

  - Línea 1 del fichero: Valores n y g, en este orden.

[1] En el grafo, los vértices son las aldeas y las aristas las carreteras; aunque no necesariamente el grafo dispondrá de todas sus aristas, se garantiza que todos los vértices son alcanzables entre sí (grafo conexo).

- Línea 2: Censo de vehículos: Una lista de n números enteros mayores que cero.
- Línea 3 y $n-1$ líneas siguientes: Matriz simétrica que contiene el camino más corto entre cualquier par de aldeas según se ha descrito anteriormente.

Por tanto, el fichero contendrá $n+2$ líneas que finalizarán con un salto de línea, salvo en todo caso, la última línea. El carácter separador entre números siempre será el espacio en blanco.

A través de *Moodle* se puede descargar un archivo comprimido con varios ejemplos y sus soluciones, entre ellos está `07a03g.p` utilizado como ejemplo en este enunciado.

- **Salida del programa, formato de salida y ejemplo de ejecución.**

  El programa mostrará cuatro resultados:

  1. Coste asociado a la ubicación escogida, precedido de la etiqueta "`Backtracking: `". Por ejemplo: `Backtracking: 207.5`
  2. Relación de aldeas en las que se propone la instalación de gasolinera, utilizando el carácter blanco como separador y precedida de la etiqueta "`Emplacements: `". El orden en el que se muestran las aldeas con gasolinera es indiferente.
     Por ejemplo: `Emplacements: 3 4 5`
  3. Tiempo de CPU (en milisegundos) requerido para encontrar la solución, precedido de la etiqueta "`CPU time (ms): `". Por ejemplo: `CPU time (ms): 0.095262`.
     Debe utilizarse la librería `chrono` tal y como se ha hecho en prácticas anteriores.
  4. Número de llamadas recursivas que el algoritmo de *backtracking* realiza, precedido de la etiqueta "`Recursive calls: `". Por ejemplo: `Recursive calls: 128`.

  De esta manera, un ejemplo de salida del programa es la siguiente:

```
$mca_bt -f 07a03g.p
Backtracking: 207,5
Emplacements: 3 4 5
CPU time (ms): 0.095262
Recursive calls: 128
$
```

  Es imprescindible ceñirse al formato y texto de salida mostrado, incluso en lo que se refiere a los saltos de línea o carácter separador, que en todos los casos es el espacio en blanco (aunque no hay problema si hay más de uno). La última línea debe terminar con un salto de línea (y sólo uno). **Debe respetarse los textos y formatos mostrados y en ningún caso debe añadirse texto o valores adicionales.** Deberá tratarse también posibles errores en los argumentos de la orden, no suministrar el fichero de entrada o su inexistencia, tal y como se hizo en las prácticas anteriores.

- **Sobre la evaluación de esta práctica.**

  Además del uso apropiado de la estrategia y de la obtención del resultado correcto, en esta práctica se tendrá en cuenta también el tiempo de CPU que se requiere para obtener la solución. En este sentido puede ser relevante:

  1. Forma de expandir el árbol de estados posibles.
  2. Uso de un subóptimo de partida lo más ajustado posible (cota pesimista del estado inicial).
  3. Uso de técnicas de poda basadas en la mejor solución hasta el momento (cotas optimistas).

Normas para la entrega.

> **ATENCIÓN: Estas normas son de obligado cumplimiento para que esta práctica sea evaluada.**

1. Se debe entregar únicamente el código fuente, con nombre `mca_bt.cc`, y el fichero `makefile`. No hay que entregar nada más, en ningún caso se entregarán ficheros de test.
2. Es imprescindible que no presente errores ni de compilación ni de interpretación (según corresponda), en los ordenadores del laboratorio asignado.[2] Se tratará de evitar también cualquier tipo de *warning*.
3. Todos los ficheros que se entregan deben contener el nombre del autor y su DNI (o NIE) en su primera línea (entre comentarios apropiados según el tipo de archivo).
4. Se comprimirán en un archivo *.tar.gz* cuyo nombre será el DNI del alumno, compuesto de 8 dígitos y una letra (o NIE, compuesto de una letra seguida de 7 dígitos y otra letra). Por ejemplo: 12345678A.tar.gz o X1234567A.tar.gz. **Solo se admite este formato de compresión y solo es válido esta forma de nombrar el archivo.**
5. En el archivo comprimido **no debe existir subcarpetas**, es decir, al extraer sus archivos estos deben quedar guardados en la misma carpeta donde está el archivo que los contiene.
6. La práctica hay que subirla a *Moodle* respetando las fechas expuestas en el encabezado de este enunciado.

[2] Si trabajas con tu propio ordenador o con otro sistema operativo asegúrate de que este requisito se cumple.