ANÁLISIS Y DISEÑO DE ALGORITMOS

Branch & Bound

Práctica 8 de laboratorio

Entrega: Hasta el domingo 3 de Junio de 2018, 23:55h (a través de Moodle)

Asignación de coste mínimo IV

`El enunciado del problema es idéntico al de la práctica anterior pero en este caso hay que resolverlo mediante la técnica` *`Branch & Bound`*`.`

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 ***Branch & Bound*** (ramificación y poda) para obtener la mejor ubicación de las gasolineras.

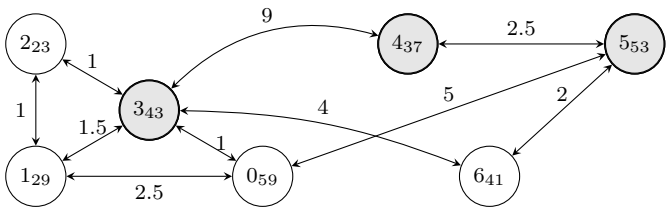

Figura 1: Posible ubicación de las tres gasolineras en los vértices sombreados en gris. Los habitantes de las aldeas 0, 1 y 2 irían a repostar a la aldea 3 (ya que es donde está la gasolinera mas cercana), y los de la aldeas 6 irían a la 5. Los de las demás aldeas no se moverían ya que dispondrían de 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_bb**. La orden tendrá la siguiente sintaxis:

mca_bb -f `fichero_entrada`

[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).

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.
- 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 "`Bb: `".
  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.
  3. Tiempo de CPU (en milisegundos) requerido para encontrar la solución, precedido de la etiqueta "`CPU time (ms): `". Debe utilizarse la librería `chrono` tal y como se ha hecho en prácticas anteriores.
  4. Número de iteraciones que realiza el bucle *while* de ramificación y poda, precedido de la etiqueta "`Iterations of loop while: `".

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

```
$mca_bb -f 07a03g.p
Bb: 207,5
Emplacements: 3 4 5
CPU time (ms): 0.095262
Iterations of loop while: 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 de la implementación del algoritmo de búsqueda 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 los mecanismos y estrategias empleados para acelerar la búsqueda. En concreto:

  1. La forma en la que se expanden los nodos internos así como el orden en el que se extraen los elementos de la lista de nodos vivos, es decir, la estrategia de búsqueda.
  2. El uso de cotas optimistas y cotas pesimistas.

- **Documentación del trabajo realizado.**
  Se debe entregar una **memoria**, en formato pdf y con nombre `memoria.pdf`, que describa las estructuras de datos empleadas así como la/s estrategia/s de búsqueda, los mecanismos de poda y las cotas optimistas y pesimistas utilizadas. Opcionalmente, puede realizarse también un análisis del comportamiento de distintas estrategias de búsqueda en conjunción con distintos mecanismos de poda. Para ello, se mostrará un cuadro comparativo con el número de iteraciones que realiza el bucle principal en cada caso estudiado.

  La primera línea de la memoria contendrá el nombre del autor y su DNI. El resto se estructurará en seis apartados de la siguiente manera:

  1. **Estructuras de datos**
     1.1 **Nodo**
     En este apartado se describirá brevemente el contenido del nodo, es decir, las variables que lo componen y su cometido.
     1.2 **Lista de nodos vivos**
     Breve descripción de la estructura de datos utilizada y si procede, qué criterio se sigue para extraer el nodo más prometedor (si se han analizado varios criterios, se mencionará en este apartado el que se ha considerado mejor).
  2. **Mecanismos de poda**
     2.1. **Poda de tuplas no factibles**
     Se deberá indicar cómo se descartan las nodos que no son factibles, si no procede este tipo de poda debe indicarse el motivo.
     2.2. **Poda de tuplas no prometedoras**
     Se deberá indicar cómo se decide que un nodo no es prometedor, si no procede debe indicarse el motivo.
  3. **Cotas pesimistas y optimistas**
     Se describirá en qué consisten dichas cotas, distinguiéndose entre el nodo inicial y los demás nodos.
     3.1 **Cota pesimista y optimista del nodo inicial**
     3.2 **Cota pesimista y optimista de los demás nodos**
  4. **Otros medios empleados para acelerar la búsqueda**
     En este apartado se citará cualquier otro mecanismo utilizado con el objetivo de acelerar la búsqueda.
  5. **Estudio comparativos de distintas estrategias de búsqueda**
     Si opcionalmente se han analizado distintas estrategias de búsqueda en conjunción con distintos mecanismos de poda, se mostrará, para cada caso estudiado, un cuadro comparativo con el número de iteraciones que realiza el bucle principal y el número de nodos que se han podado (tal y como se ha explicado en algunos ejemplos de teoría).
  6. **Soluciones y tiempos de ejecución**.
     Se mostrará en este apartado las soluciones obtenidas y el tiempo de proceso requerido para cada uno de los ficheros de test publicados para esta práctica. Esta información se indicará en una relación similar a la que se expone a continuación (como es lógico los tiempos de proceso dependen de la calidad de la implementación, del ordenador, etc.) Si alguno de los ficheros no ha podido ser resuelto por el programa realizado se escribirá '¿?' en el lugar correspondiente a su solución (como aquí se muestra en el último fichero).
     - Fichero 10a-05g.p: Solución: xxx; Tiempo de proceso: yyy ms
     - Fichero 20a-10g.p: Solución: xxx; Tiempo de proceso: yyy ms.
     - Fichero 30a-12g.p: Solución: xxx; Tiempo de proceso: yyy ms.
     - Fichero 30a-25g.p: Solución: xxx; Tiempo de proceso: yyy ms.
     - Fichero 40a-10g.p: Solución: xxx; Tiempo de proceso: yyy ms.
     - Fichero 40a-30g.p: Solución: ¿?

IMPORTANTE: Los apartados 1 al 4 se complementarán con el correspondiente fragmento de código que corrobore el trabajo descrito. Si algunos de los apartados no ha sido realizado se pondrá únicamente la frase "NO IMPLEMENTADO".

De los distintos elementos que pueden contener un algoritmo de Ramificación y poda, sólo se evaluarán los que estén reflejados en esta memoria. Es decir, todo lo que no esté en el documento se entenderá que no ha sido realizado.

Cabe destacar que se puede optar a la máxima nota aún sin implementar todos los apartados de esta memoria pues pueden existir estrategias de búsqueda que hagan innecesario el empleo de alguno de los mecanismos que tratan aceleran la búsqueda.

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_bb.cc`, el fichero `makefile` y `memoria.pdf`. 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.