

PRÁCTICA 4 – Backtracking y Branch-and-Bound. Un repartidor muy eficaz

Estructuras de datos y Algoritmos II (EDA II). 2º Grado en Ingeniería Informática. Universidad de Almería

Version curso.2022

Objetivos

- Construir soluciones a un problema utilizando los **métodos algorítmicos *backtracking*** (vuelta atrás) y ***branch-and-bound*** (ramificación y poda). Analizar y comparar los algoritmos implementados desde diferentes perspectivas.
- Realizar el **análisis de la eficiencia** de las soluciones aportadas, y una comparativa tanto desde el punto de vista teórico como práctico.

Requerimientos

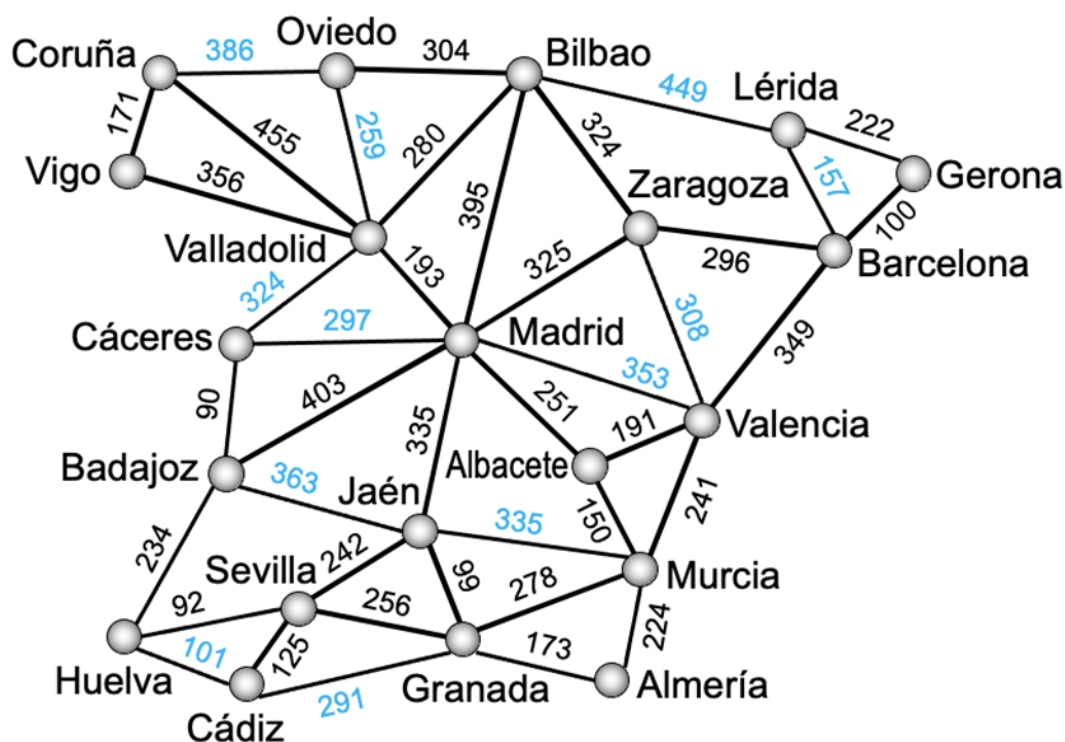
Para superar esta práctica se debe realizar lo siguiente:

- Recordar (de EDA I) las **estructuras de datos** más apropiadas para la resolución de problemas.
- Conocer cómo implementar los esquemas algorítmicos ***backtracking*** y ***branch-and-bound*** para responder a una necesidad concreta (en este caso, resolver el problema del viajante de comercio y de ciclos hamiltonianos).
- **Evaluar los algoritmos implementados** siguiendo los métodos ***backtracking*** y ***branch-and-bound***, relacionando ambos métodos algorítmicos.

Enunciado del problema

Como sabemos, en EDAland existen ciudadanos muy característicos, entre ellos, nos encontramos a ‘Braulio el repartidor’, un trabajador incansable y responsable de Almería que lleva toda una vida repartiendo los periódicos del Hoy-Almeria por todo el país. Todas las madrugadas, cuando la editorial del Hoy-Almeria cierra el diario y salen los primeros ejemplares de la imprenta, Braulio está ahí, con su furgoneta, listo para repartirlos entre todas las ciudades de EDAland. Además de por su eficacia repartiendo, Braulio también destaca por ser una persona que tiene muy en cuenta su economía, y siempre procura escoger el circuito

que le suponga recorrer la distancia más corta posible o realizar el menor gasto de combustible (y más ahora, al elevado precio que está) posible. Para ser tan eficaz, el bueno de Braulio se planteó tres cuestiones importantes. La primera consistía en conocer todos los posibles circuitos, partiendo desde Almería, que recorren cada ciudad, donde debe dejar un lote de periódicos, exactamente una vez y regresando posteriormente a Almería. La segunda cuestión consistía en determinar un circuito que recorra cada ciudad exactamente una vez, partiendo y regresando a Almería, y habiendo recorrido en total la menor distancia posible. La tercera y última cuestión consistía en determinar un circuito que recorra cada ciudad exactamente una vez, saliendo y volviendo a Almería y cuyo gasto en combustible sea el mínimo posible.



Una vez comprobado el correcto funcionamiento de nuestros algoritmos en la red reducida de caminos, analizaremos nuestros algoritmos en la **red nacional de carreteras** de EDALand (Figura 2, adaptado del grafo de ciudades y carreteras de [1]), que conecta a todos los núcleos de población del país (1053 núcleos de población con 2017 carreteras con la distancia que hay entre dos poblaciones). Ésta es la misma red de la práctica 2.



Figura 2. Red nacional de caminos de EDAland

Trabajo a desarrollar

Deberá proponer e implementar soluciones con los esquemas algorítmicos de **backtracking** y **branch-and-bound**, según se requiera, a los problemas que se plantean a continuación.

- (**Backtracking**) Determinar todos los posibles circuitos, si es que hubiera más de uno, partiendo desde Almería, que recorren cada ciudad de la nueva red reducida de carreteras de EDAland, donde Braulio debe dejar un lote de periódicos, exactamente una vez y regresar a Almería.
- (**Backtracking**) Determinar un circuito que, partiendo desde Almería, visite cada ciudad exactamente una vez, regresando a Almería y habiendo recorrido en total la menor distancia posible. Resolver este problema para la nueva red reducida de carreteras de EDAland.
- (**Branch-and-Bound**) Modificar el valor de las aristas de la nueva red reducida de carreteras de la siguiente manera (esto supone que tendremos una nueva red reducida de carreteras con valores diferentes en las aristas): multiplicar el valor de la distancia por un número real aleatorio x , $0.5 \leq x \leq 1.0$, de tal forma que el peso de las aristas ahora sea un número real (**double**) con dos cifras decimales, que podría corresponderse con el gasto en combustible de desplazarse de una ciudad a otra. Sobre esta nueva red, determinar un circuito que debe seguir Braulio, partiendo de Almería, recorra cada ciudad exactamente una vez, regrese a Almería y cuyo gasto en combustible sea el mínimo posible.
- Haciendo uso de los algoritmos (**Backtracking** y **Branch-and-Bound**) implementados en los apartados anteriores, debe intentar obtener el circuito en la red nacional de carreteras

completa, partiendo de un núcleo urbano cualquiera, que visite cada población exactamente una vez, regrese al núcleo de partida y tenga la menor distancia posible. ¿Qué conclusiones obtiene del intento?. ¿Existe alguna forma de resolver el problema que se le ha planteado?. Indíquelo todo de forma razona, justificándolo correctamente. Además, para comprobar que los algoritmos funcionan sobre dicha red, lleve a cabo una traza de la ejecución de los mismos en la que muestre su estado en función de la iteración o cada cierto tiempo.

Para ello deberá realizar los siguientes apartados:

- **Estudio de la implementación:** Explicar los detalles más importantes de la implementación llevada a cabo, tanto de las estructuras de datos utilizadas para resolver el problema en cuestión, como de los algoritmos implementados. El código debe de estar razonablemente bien documentado (**JavaDoc**).
- **Estudio teórico:** Estudiar los tiempos de ejecución de los algoritmos de *backtracking* y *branch-and-bound* implementados, en función del número de núcleos de población (vértices) y del número de carreteras (aristas). Comparar también los algoritmos propuestos, teniendo en cuenta las características de la red (grafo) y las técnicas de implementación elegidas.
- **Estudio experimental:** Validación de los algoritmos de *backtracking* y *branch-and-bound* implementados sobre las redes de EDAland proporcionadas. Para ello, se deberán obtener y comparar los tiempos de ejecución de los algoritmos implementados. Se contrastarán los resultados teóricos y los experimentales, comprobando si los experimentales confirman los teóricos previamente analizados. Se justificarán los experimentos realizados, y en caso de discrepancia entre la teoría y los experimentales obtenidos se deberá intentar buscar una explicación razonada.

Entregas

Se ha de entregar, en fecha, un repositorio [privado](#) de GitHub con acceso para el/los profesor/es que evalúa/n las prácticas (mismo repositorio para todas las prácticas de EDA II), con toda la documentación y el código fuente requerido en la práctica:

- En dicho repositorio debe estar la carpeta llamada **practica_4**, donde debe haber como mínimo una carpeta **src** para el código fuente, y una carpeta **docs** para la documentación, siguiendo la estructura de proyecto Java descrita en la práctica anterior.
- Memoria que explique todo lo que habéis realizado en la práctica. La memoria deberá tener el formato que se indica a continuación. Si se desea, también se podrá realizar una presentación de la práctica.
- Código fuente de la aplicación, desarrollada en JAVA, que resuelva todo lo planteado en la práctica. Recordad que tendréis que medir tiempos de ejecución de vuestras soluciones por lo que deberéis incluir las órdenes necesarias para ello en el código fuente.
- Juegos de prueba que consideréis oportunos incluir para asegurarnos de que todo funciona correctamente (en la carpeta **test**).

La memoria de práctica a entregar debe ser breve, clara y estar bien escrita. Ésta debe incluir las siguientes secciones:

- Una breve **introducción** con un estudio teórico del método algorítmico utilizado en esta práctica (*backtracking* y *branch-and-bound*).
- Una sección para cada uno de **apartados propuestos** a desarrollar en esta práctica (estudio de la implementación, estudio teórico y estudio experimental). Hemos de remarcar que deben incluirse los apartados en el mismo orden en el que se han expuesto.
- Se incluirá también un **anexo** con el diseño del código implementado con diagramas de clases y cualquier otro diagrama que estiméis necesario incluir, no introducir ningún código en este anexo. Además, añadir en esta sección una lista de los archivos fuente y una breve descripción del contenido de cada uno.
- Es importante incluir siempre las **fuentes bibliográficas** utilizadas (web, libros, artículos, etc.) y hacer referencia a ellas en el documento.

Evaluación

Cada apartado se evaluará independientemente, aunque es condición necesaria para aprobar la práctica que los programas implementados funcionen correctamente.

- La implementación junto con la documentación del código se valorará sobre un 40%
- El estudio de la implementación se valorará sobre un 10%
- El estudio teórico se valorará sobre un 15%
- El estudio experimental se valorará sobre un 35%

Se penalizará no entregar el apartado de introducción teórico o una mala presentación de la memoria.

Se podrá requerir la defensa del código y de la memoria por parte de profesor.

Fecha de entrega

Fecha de entrega: 5 de Junio de 2022

Referencias

- [1] Gines García Mateos. El Reto del Viajante. Disponible online en <http://dis.um.es/~ginesgm/retoviajante.html> [Fecha de consulta: 19/03/2022]