PRÁCTICA 3 – Programación Dinámica: Una decisión bien tomada, un tesoro bien escogido

Estructuras de datos y Algoritmos II. Universidad de Almería

Version **curso.2022**

Objetivos

- Construir soluciones a un problema utilizando el **método algorítmico de programación dinámica** (o dynamic programming). 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 el **esquema de programación dinámica** para responder a una necesidad concreta (en este caso, resolver un problema de optimización).
- **Evaluar los algoritmos de programación dinámica implementados**, y compararlos en su caso con implementaciones de algoritmos greedy, relacionando ambos métodos algorítmicos.

Enunciado del problema

En EDAland existen personajes muy variopintos, entre ellos, nos encontramos a ‘Pedrito el curioso’, un intrépido aventurero de Almería que desde su más tierna infancia quería emular a Indiana Jones, devorando libros en la biblioteca sobre misterios sin resolver, leyendas de ciudades fantasma y tesoros perdidos. Después del último gran terremoto acaecido en EDAland de una magnitud de 7,3 grados en la escala de Richter, que se notó hasta en su país vecino InSoland, ha modificado parte de lo orografía del país y ha abierto grandes brechas en la cordillera suroriental. Este hecho ha disparado la chispa aventurera de Pedrito, que guiado por su conocimiento sobre tesoros ocultos y por su valentía a la hora de emprender retos de espeleología, se ha embarcado en la difícil

misión de explorar la gruta del Zorokotroko, pico más alto de esa cordillera, y donde cuenta la leyenda que está escondido el gran tesoro del pirata Zrokk el Sanguinario.

La misión es bastante complicada y arriesgada ya que, a lo largo del tiempo, varios grupos especializados han intentado explorarla sin éxito, pues más allá de 500 metros de la entrada existe una peligrosa caída de más de 1000 metros, cuyo fin parece estar en el centro de la Tierra. Pedrito, que no tiene miedo a nada, se embarca en una nueva aventura, y ataviado con su equipo de espeleología se adentra en la gruta con el objetivo de encontrar el tesoro. Éste no sigue el camino marcado por los anteriores grupos que exploraron la cueva, si no que se desvía por una nueva grieta abierta por el terremoto y que es totalmente nueva. Después de andar varias centenas de metros hacia abajo por un abrupto túnel de no más de un metro de altura, Pedrito descubre sorprendido que, al enfocar su potente linterna, a lo lejos se divisa algo que brilla. Sigue bajando unas cuantas decenas de metros más, y sus primeras sospechas se hacen realidad, descubre una pequeña cavidad (cueva) donde se encuentran perfectamente ubicados varios objetos que brillan al proyectarle luz y da la sensación de que pueden ser de gran valor. La primera sensación de Pedrito es que … encontró el tesoro de Zrokk, ¡¡EUREKA!!

Cuando Pedrito, por fin, consigue llegar al lugar donde está todo el tesoro se da cuenta de que no son muchos los objetos que allí hay, pero sí puede apreciar que todos ellos son muy valiosos. Dado su gran conocimiento sobre tesoros y al sopesar todos los objetos puede tener una muy buena idea del peso y valor de cada uno de los objetos que conforman el tesoro de Zrokk.

Debido a lo peligroso que es el camino de vuelta (hay ruidos en las rocas muy sospechosos de que puede producirse un derrumbe de un momento a otro) y lo limitada que es la capacidad de su mochila de espeleólogo, Pedrito se enfrenta con la difícil tarea de que solo puede llevar consigo aquellos objetos que quepan en su mochila, que tiene una capacidad (peso que soporta) limitada.

Figura 1. La mochila de Pedrito y el tesoro de Zrokk.

Trabajo a desarrollar

Deberá proponer e implementar soluciones con el esquema algorítmico de programación dinámica a los problemas que se plantean a continuación.

- Suponiendo que los objetos no se pueden romper y que los pesos son exactos (número naturales positivos), determinar qué objetos debe elegir Pedrito para maximizar el valor total de lo que pueda llevarse en la mochila, sabiendo que los valores de los objetos que estima Pedrito son números reales positivos.
- Suponiendo que justo al lado de esa cueva, Pedrito descubre que hay otra mucho más grande, cuyos objetos no alcanza a contar. Lo que si puede observar es hay una cantidad inagotable de objetos preciosos de tipos (clases) distintos. Cada objeto de un tipo es indivisible (no se puede romper), tiene un cierto peso (natural positivo) y un cierto valor (real positivo) que Pedrito conoce perfectamente. Determinar qué cantidad de objetos de cada tipo debe coger Pedrito para maximizar el valor total de lo que puede llevarse en su mochila.
- Suponiendo que los objetos no se pueden romper, pero los pesos pueden no ser exactos (los pesos pueden ser número reales positivos). Indicar cómo habría que modificar el algoritmo del primer aparatado para que Pedrito se pudiera llevar en su mochila el máximo de riquezas.
- Imaginemos el caso de que los objetos valiosos que encuentra Pedrito en la cueva se pueden romper (fraccionar), ¿sería sencillo encontrar una solución con *programación dinámica* para que Pedrito determine qué objetos elegir para maximizar el valor total de lo que puede llevarse en su mochila?. ¿Sería más sencillo realizarlo con un algoritmo **greedy** (voraz)?. De ser así, implementarlo y exponer las principales diferencias entre los dos esquemas algorítmicos (programación dinámica y greedy) para este caso concreto.

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 programación dinámica (PD) implementados, en función del número de objetos y sus valores. Comparar también los algoritmos propuestos, teniendo en cuenta las características del problema concreto que resuelven y cómo se actualizan las tablas utilizadas para ello. Comparar las dos técnicas utilizadas: programación dinámica y greedy, destacando sus características, ventajas, desventajas, etc.
- **Estudio experimental**: La validación de los algoritmos de PD implementados para resolver los problemas planteados, pueden utilizar los conjuntos de datos disponibles

en la siguiente Web: https://people.sc.fsu.edu/~jburkardt/datasets/knapsack_01/knapsack_01.html. Podemos ver que existen varios conjuntos, y en ellos se pueden observar datos relativos a: capacidad de la mochila, pesos de los objetos, valor o beneficio de los objetos y la selección óptima de pesos. Para ello, se deberán comprobar el correcto funcionamiento y, 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 experimentos se debe intentar buscar una explicación razonada. Además, se generarán **datos aleatorios**, en concordancia con el formato de los disponibles en la Web, para los casos de tipos/clases distintos de objetos (número inagotable de objetos en cada tipo/clase) y para el caso de que los objetos se puedan fraccionar (para este caso, tener en cuenta algún conjunto de la Web). Para este último caso, se probarán los datos con algoritmo greedy implementado.

Entregas

Se ha de entregar, en fecha, un repositorio privado de GitHub con acceso para el profesor que evalúa 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 crear una nueva carpeta llamada `practica_3`, donde creéis dos subcarpetas una para la documentación, `docs` y otra para el código fuente `sources`.
- 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.

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 (programación dinámica).
- 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: **08 de Mayo de 2022**