UNIVERSIDAD DE GRANADA

E.T.S. DE INGENIERÍAS INFORMÁTICA y DE TELECOMUNICACIÓN

Departamento de Ciencias de la Computación e Inteligencia Artificial

Algorítmica

Guión de Prácticas

Práctica 1: Divide y Vencerás

Curso 2023-2024

Grado en Informática

Índice

Objetivo

El objetivo de esta práctica es que el estudiante comprenda y asimile el funcionamiento de la técnica de diseño de algoritmos Divide y Vencerás. Esta práctica será realizada por grupos de alumnos. Cada grupo tendrá que diseñar varios algoritmos según las indicaciones incluidas en este guión.

1. Umbrales en la técnica "Divide y vencerás"

La idea fundamental de técnica 'Divide y vencerás' es la descomposición de un problema dado en problemas, habitualmente del mismo tipo, pero de complejidad (tamaño) menor. Esto se suele traducir en un proceso recursivo que requiere de alguna condición de parada, por debajo de la cual ya no se aplica recursividad sino que se resuelven directamente los casos de problema que han surgido. Eso requiere que, además, del método general, se disponga de un algoritmo específico para resolver los casos más pequeños. A este algoritmo se le denomina algoritmo base o específico.

Una cuestión clave es determinar cuál es el criterio para dejar de usar la recursividad y aplicar el método específico directamente. Este criterio se implementa definiendo un valor para el tamaño del problema a partir del cual se resuelve con el método específico. A este tamaño se le denomina **umbral** y su determinación es un aspecto crítico en la implementación del algoritmo "Divide y vencerás". Sabemos que su valor óptimo no es fijo, y que depende de diversos factores: algoritmo base, implementación, casos estudiados, etc.

En esta práctica vamos a abordar su cálculo según los tres enfoques que se describen a continuación.

1. Cálculo del valor del umbral teórico. A partir de la expresión del tiempo de ejecución del algoritmo base (teórica) y la expresión recurrente del tiempo de ejecución para el método recursivo, hay que determinar cuál es el tamaño del problema para el que ambos tiempos son iguales (suponiendo un sólo nivel de recursión). Es decir, se plantea una ecuación igualando el tiempo del algoritmo específico al del desarrollo en un sólo paso del tiempo del algoritmo recursivo. Se resuelve esta ecuación y este valor es el umbral teórico. Este resultado tiene un valor normalmente indicativo dado que algoritmo específico se aplica para casos de tamaño pequeño, y en esta región las constantes ocultas tienen un papel importante.

2. Cálculo del valor del umbral óptimo. Está definido para cada implementación concreta del algoritmo. Se necesita conocer el tiempo exacto según un enfoque híbrido del tiempo de ejecución del algoritmo específico. Se plantea una ecuación igualando esta fórmula de tiempo con la expresión recurrente del método recursivo. Y se resuelve la ecuación resultante.

3. Umbrales de tanteo. Se evalúan los tiempos de ejecución resultantes para distintos valores (inferiores y superiores al umbral óptimo).

4. Cálculo gráfico del umbral. Representar en una gráfica comparativa la evolución del tiempo de ejecución del algoritmo para los distintos valores del umbral estudiados. Es decir,

para cada selección distinta del umbral (teórico, óptimo y de tanteo) deberá dibujar una curva con los tiempos de los casos medidos empíricamente. Comente los resultados justificando cuál o cuáles de las variantes darían mejores resultados.

2. Enunciado de los problemas

En esta sección se incluyen los enunciados de los problemas que tienen que resolver cada grupo de alumnos. Los problemas considerados son los que se describen en las subsecciones de esta sección

2.1. P1: Subsecuencia de suma máxima

Dado una secuencia de valores reales (positivos o negativos), almacenada en un vector, se pide encontrar una subsecuencia de elementos consecutivos que cumpla que la suma de los elementos de la misma sea la máxima posible.

Aunque existe el algoritmo de Kadane que permite encontrar de forma óptima el valor de dicha subsecuencia en $\Theta(n)$, se pide encontrar un algoritmo que siguiendo la estrategía divide y vencerás encuentre una solución al problema lo más eficiente posible, esto es, con dicho orden de eficiencia.

2.2. P2: Enlosar un espacio

Queremos cubrir el suelo de una habitación cuadrada con baldosas en forma de `'ele'`. En el suelo sabemos de la ubicación de un sumidero que ocupa exactamente una loseta. Se pide encontrar la forma de colocar las losetas (sin romper ninguna) en nuestro suelo.

Asumiremos que el suelo se representa como una matriz bidimensional A de tamaño $n \times n$, donde $n = 2^k$ para algún $k >= 1$. La posición del sumidero la conocemos a priori por el par de valores i, j con $0 \leq i, j \leq n - 1$. En nuestra matriz almacenaremos en la celda $A[i][j]$ el valor 0. Para cada baldosa que coloquemos en la matriz le asociaremos un identificador (entero) distinto.

Se pide diseñar una algoritmo que permita encontrar la forma de rellenar el suelo (la matriz) de baldosas.

Por ejemplo, para una entrada del tipo

```
n = 2
i = 0
j = 0
Salida:
0       1
1       1

n = 4
i = 0
j = 0
Salida:
```

```
0       3       2       2
3       3       1       2
4       1       1       5
4       4       5       5
```

2.3. P3: Problema del viajante de comercio

Tenemos un conjunto de n ciudades (puntos en un plano), cada una definida por las coordenadas en el mapa (x_i, y_i), con $i = 1, \ldots, n$. La distancia entre dos ciudades viene dada por la distancia euclídea entre sus coordenadas.

$$dist((x_1, y_1), (x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

El problema de viajante de comercio consiste en encontrar el orden en el que un viajante, partiendo de la ciudad de origen (por ejemplo (x_1, y_1)) pase por todas y cada una de las ciudades una única vez, para volver a la ciudad de partida, formando un ciclo.

El costo del ciclo será la suma de las distancias que hay entre las ciudades.

El problema original del viajante de comercio consiste en encontrar el ciclo de costo mínimo entre todas las posibilidades existentes.

Aunque este problema es NP-Difícil y por tanto no podemos esperar encontrar una solución óptima al mismo, lo que se pretende en esta práctica es utilizar la estrategía del divide y vencerás para encontrar una solución aproximada que puede ser de utilidad en situaciones como las que se plantea en este problema.

3. Trabajo a realizar

Para alcanzar el objetivo previsto con esta práctica, los alumnos deberán realizar las tareas que se detallan en la sección siguiente, documentando el trabajo realizado en una memoria con el contenido y estructura que se indica en la sección 4.

3.1. Tareas

1. Estudiar en profundidad cada problema, asegurándose de comprender bien las entradas y salidas involucradas, así como las relaciones entre éstas. Deberán identificar el tipo de información que define cada caso del problema y las magnitudes que definen su tamaño.

2. Diseñar e implementar un generador de casos para el problema.

3. Diseñar un algoritmo específico para resolver el problema, que se aplicará en la resolución de aquellos casos cuyo tamaño sea inferior al umbral.

4. Diseñar un algoritmo para resolver el problema basado en la técnica “Divide y vencerás”.

5. Implementar tanto el algoritmo específico como el basado en “Divide y vencerás”.

6. Realizar el análisis de la eficiencia teórica, empírica e híbrida de ambos algoritmos.

7. Calcular los umbrales (teórico, óptimo y de tanteo) para las implemen- taciones de algoritmos realizadas.

8. Elaborar una memoria que documente el trabajo realizado de acuerdo a las indicaciones incluidas en la sección 4.

4. Memoria

Todo el trabajo realizado debe redactarse en una memoria. La estructura de este documento será la siguiente:

1. Portada. Incluyendo las denominaciones de titulación, asignatura y práctica. También el nombre completo de los alumnos que forman el grupo y su dirección de correo electrónico.

2. Autores. Indicar el % del trabajo realizado por cada alumno, especificando qué tareas ha realizado cada uno.

3. Objetivos. Descripción del objetivo de la práctica.

4. Definición del problema. Descripción de los casos usados en la evaluación de la eficiencia. Descripción completa del entorno de análisis: hardware, sistema operativo, compilador, etc. empleados. Descripción del método de medición de tiempos.

5. Algoritmo específico. Diseño del algoritmo. Detalles de implementación. Análisis de la eficiencia teórica, empírica e híbrida

6. Algoritmo divide y vencerás. Diseño del algoritmo. Detalles de implementación. Cálculo de umbrales teórico, óptimo y de tanteo. Análisis de la eficiencia teórica, empírica e híbrida.

7. Conclusiones.

5. Evaluación de la práctica

Esta práctica se realizará por grupos. Los alumnos habrán de entregar dos archivos en la actividad correspondiente incluida en la página de la asignatura en la plataforma Prado.

- El primer archivo será la memoria, en formato pdf.

- El segundo será el código fuente de las implementaciones realizadas empaquetado en un fichero .zip. La Figura 1 ilustra la estructura que se debe crear donde para cada problema estudiado tendremos una carpeta distinta, llamadas P1, P2 y P3. En cada carpeta el código estará organizado en tres subcarpetas de nombres: Generador, Epecifico y Dyv. Cada una de esas carpetas incluirá todos los módulos de código fuente necesarios para generar el programa binario correspondiente las respectivas implementaciones de: generador de casos del problema, algoritmo específico y algoritmo basado en “Divide y vencerás”.

Figura 1: Estructura para entregar en prácticas

```
P1
  |- Generador
  |- Especifico
  |- DyV
  |- Instancias
     | - ni1.txt
     | - ni2.txt
     | - ...
  |- Makefile
P2
  | ...
P3
  | ...
```

El generador de casos del problema deberá almacenar las instancias que cree en el directorio Instancias. Dicho programa deberá aceptar como entradas al menos dos parámetros, el primero el tamaño de la instancia a crear y el segundo el nombre del fichero salida para dicha instancia. Se generará una instancia y el resultado se almacenará en el directorio instancias con en nombre indicado.

Los programas especifico y dyv aceptarán un único argumento en línea de órdenes, que será el nombre de un fichero que incluirá los datos de un caso que trata de resolver y enviarán a la salida estándar la solución calculada.

Además, en la carpeta raíz habrá un único fichero Makefile cuyo objetivo por defecto construirá los tres binarios de nombres generador, especifico y dyv, respectivamente.

La fecha límite para entregar la memoria es el día 9 de abril de 2023 a las 23:59 horas. Además de hacer el trabajo y entregar la memoria, cada equipo tendrá que elaborar una breve presentación del trabajo realizado que expondrá públicamente en clase de prácticas, de acuerdo a la convocatoria establecida por el profesor. Es **obligatoria** la participación de todos los miembros del grupo en la exposición.