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:
Técnica Divide y Vencerás

Curso 2011-2012

Grado en Informática

Práctica 1

Técnica Divide y Vencerás

1. Objetivos y Evaluación

El objetivo de la práctica es que el alumno comprenda y asimile el funcionamiento de la técnica de diseño de algoritmos "Divide y Vencerás". Esta técnica consiste en resolver un problema a partir de la solución de subproblemas del mismo tipo, pero de menor tamaño. Para ello se propone un problema que debe ser resuelto utilizando algoritmos basados en dicha técnica y sobre el que se realizará un estudio de su eficiencia teórica, empírica e híbrida.

La práctica se evalúa sobre **1 punto**. Se valorará tanto la correcta implementación de los algoritmos como el análisis posterior. La no consideración de las especificaciones de formato y demás cuestiones expresadas en este guión podrán influir negativamente en la evaluación de la práctica.

2. Multiplicación de Enteros Largos

2.1. Descripción del Problema

El problema que consideraremos en cómo realizar operaciones aritmticas con enteros con una gran cantidad de dígitos, particularmente la multiplicación. Este tipo de operaciones son importantes, entre otras, en aplicaciones criptográficas, donde la eficacia de los sistemas depende de su capacidad para hacer cálculos con enteros con cientos de dígitos. La suma y la resta de éstos números es sencilla, pudiéndose ejecutar en $O(n)$, siendo n el número de digitos. Sin embargo, el uso de un algoritmo clásico de multiplicación requerirá un tiempo en el $O(n^2)$, que puede resultar excesivo cuando necesitamos un alto número de multiplicaciones.

2.2. Descripción de la Técnica Divide y Vencerás

El algoritmo de multiplicación toma como entrada dos enteros largos A y B con longitud n (por simplicidad supondremos n potencia de 2) y da como salida el entero largo resultado de multiplicar $A \times B$. Asumiremos A[0] como el dígito menos signicativo y A[n-1] como el más signicativo. La

siguiente tabla muestra la representación del número 12268.

	Pos	0	1	2	3	4
12268	$\Longrightarrow$	8	6	2	2	1

Notaremos por tanto el número como A[n-1::0] en lugar de A[0::n-1]. Sea $m = n/2$, entonces podemos definir los números w,x,y,z como números con $n/2$ cifras obtenidos mediante

w = A[n-1::m] x = A[m-1::0]
y = B[n-1::m] z = B[m-1::0]

Por tanto, podemos expresar $A = w10^m + x$ y $B = y10^m + z$ y su producto

$$A \times B = w \times y10^n + ((w \times z) + (x \times y))10^{n/2} + x \times z$$

En términos de eficiencia, la operación de multiplicar por 10^k se puede entender como simples desplazamientos de los números k posiciones a la derecha, y por tanto no se debe entender como una multiplicación (se realiza en orden $O(k)$). Además, las adiciones implican números con $n/2$ dígitos, y por tanto se realizan en un orden $O(n)$.

Por tanto, nuestro algoritmo resultante requiere 4 multiplicaciones de tamaño $n/2$ más 3 sumas y 2 desplazamientos, que están acotadas superiormente por algo de orden $O(n)$, con lo que resulta que que obtenemos la siguiente ecuación de recurrencia.

$$T(n) = \begin{cases} 1 & \text{si } n = 1 \\ 4T(n/2) + n & \text{si } n > 1 \end{cases} \tag{1.1}$$

que como sabemos, hace que nuestro algoritmo sea de un orden $O(n^2)$.

2.3. Mejorando el Algoritmo DyV

El anterior ejemplo nos permite comprender que la componente crítica del algoritmo está en el número de multiplicaciones de tamaño $n/2$ que se tienen que realizar. Como hemos visto, el tener un número constante de sumas y desplazamientos, no varía el orden de eficiencia, todas se engloban en un término $O(n)$ en la ecuación de recurrencia. Por tanto, para tener un algoritmo más eficiente, deberíamos obtener el producto con un menor número de multiplicaciones, aunque tengamos más sumas y/o desplazamientos. Este "truco" algebraico existe: Tenemos que calcular:

$w \times y$
$x \times z$
$(w \times z) + (x \times y)$

Pero en lugar de ello, podemos calcular las siguientes cantidades, pero con más sumas y restas

$C = \quad w \times y$
$D = \quad x \times z$
$E = \quad (w + x) \times (y + z) - C - D =$
$\quad (w \times y + w \times z + x \times y + x \times z) - (w \times y) - (x \times z) = (w \times z) + (x \times y)$

Con esto llegamos a poder calcular $A \times B$ de la siguiente forma,

$$A \times B = C10^n + E10^{n/2} + D$$

esto es, necesitamos 4 sumas, 2 restas de números de tamaño $n/2$ y los desplazamientos, todos ellos acotados superiormente por $O(n)$. Por tanto, el tiempo final se reduce

$$T(n) = \begin{cases} 1 & \text{si } n = 1 \\ 3T(n/2) + n & \text{si } n > 1 \end{cases}$$

(1.2)

Aplicando el Teorema Maestro, tenemos que $T(n) \in O(n^{log_2(3)}) = O(n^{1,585})$, lo que representa una mejora en el orden de eficiencia del algoritmo. Los resultados expermientales nos permitirán determinar para qué tamaños de entrada esta mejora es significativa.

3. Tareas a realizar:

El alumno dispone de la clase `enterolargo`, que define un entero largo como un vector de enteros, donde el elemento menos significativo se almacena en la posición 0, y el más significativo en la posición n-1. Se han implementado las operaciones de suma (`operator+`) y resta (`operator-`) así como algunos métodos que pueden ser de utilidad. Se debe completar la clase, con los métodos que se consideren necesarios, para permitir implementar las siguientes operaciones:

- Utilizando el TDA enterolargo, diseñar e implementar el algoritmo clásico para multiplicar dos enteros largos. Este algoritmo se implementará en el método

  ```
  enterolargo enterolargo::clasico( enterolargo & e);
  ```

  Que devuelve el resultado de multiplicar `*this` $\times$ `e`.

- Implementar el método DyVLento que multiplique dos enteros largos utilizando las 4 llamadas recursivas

  ```
  enterolargo enterolargo::DyVLento( enterolargo & e);
  ```

  Que devuelve el resultado de multiplicar `*this` $\times$ `e`.

- Implementar el método `operator*` que realiza la multiplicación siguiendo la técnica del Divide y Vencerás

  ```
  enterolargo enterolargo::operator*( enterolargo & e);
  ```

  Que devuelve el resultado de multiplicar `*this` $\times$ `e`.

- Resolver el problema usando un algoritmo Divide y Vencerás híbrido que emplee el algoritmo clásico cuando el número de dígitos es pequeño y realizar un análisis empírico del valor umbral óptimo.

  Para ello, el alumno escogerá un valor de n relativamente alto donde el tiempo de ejecución sea relativamente elevado y, por tanto, la posible ganancia de eficiencia sea significativa. Es de esperar que conforme se aumenta el valor del umbral (comenzando desde tres) se observe una ganancia de eficiencia en el algoritmo Divide y Vencerás híbrido hasta llegado un punto en el cual seguir aumentando ese valor umbral empieza a empeorar la eficiencia. Ese punto de inflexión determinará el valor umbral óptimo.

  Una vez obtenido el valor umbral óptimo, se analizará su eficiencia empírica probando con los distintos valores de n considerados en el análisis del algoritmo Divide y Vencerás puro y se comparará con éste.

4. Documentación y Entrega de la Práctica

La práctica deberá entregarse antes del **MIERCOLES 21 DE ABRIL DE 2012** . La entrega se realizará telemáticamente a través de la Web de la asignatura accesible desde

`https://decsai.ugr.es`

El nombre del fichero que contiene toda la documentación se compondrá añadiendo los apellidos y nombre del alumno separados por el símbolo de subrayado _, sin tildes y sustituyendo la letra 'ñ' por la letra 'n'. Por ejemplo, el alumno José Caños Pérez subirá el siguiente fichero: `Canos_Perez_Jose_practica_1.zip`
que contendrá los siguientes ficheros (los siguientes apartados explican el contenido de cada uno)

```
dyv_enterolargo.pdf
dyv_enterolargo.cpp
enterolargo.h
enterolargo.cpp
```

Código

Todo el código se deberá entregar en tres ficheros, `enterolargo.h` y `enterolargo.cpp`, así como el fichero donde se demuestre el correcto funcionamiento de los métodos implementados `dyv_enterolargo.cpp`.

Análisis

Una vez terminada la implementación, deberá realizarse un análisis de eficiencia teórico, empírico e híbrido. Para obtener la tabla de tiempos, el alumno deberá diseñar una batería de experimentos de la forma que estime más oportuna. Se valorará la calidad de la experimentación realizada.

Deberá entregarse un fichero pdf `dyv_enterolargo.pdf` que incluya el análisis de los resultados obtenidos para los distintos algoritmos implementados, considerando distintos tamaños de entrada.