Ejercicios en clase: División y conquista, Recurrencias

Análisis y Diseño de Algoritmos

23 de abril de 2020

Ejercicio 1. Ilustre la operación del MergeSort en el siguiente arreglo $A =< 3, 41, 52, 26, 38, 57, 9, 49 >$

Ejercicio 2. Considere la siguiente variación para insertionSort. Para ordenar el vector $A[1..n]$, ordenamos recursivamente el vector $A[1..n-1]$ y luego insertamos $A[n]$ en el arreglo ordenado $A[1..n-1]$. Escriba el pseudocódigo del algoritmo anterior. Escriba una recurrencia para el peor caso de este algoritmo. Resuelva la recurrencia.

Solucion

Recibe: un vector $A[1..n]$ y ordena el subvector $A[1..p]$

INSERTION-SORT-RECURSIVO(A, p)	*cost*	*times*
if $p > 1$	c_1	1
INSERTION-SORT-RECURSIVO$(A, p-1)$	$T(p-1)$	1
$key = A[p]$	c_3	1
$i = p - 1$	c_4	1
while $i > 0$ **and** $A[i] > key$	c_5	p
$A[i+1] = A[i]$	c_6	$p-1$
$i = i - 1$	c_7	$p-1$
$A[i+1] = key$	c_8	1

Cuando $p > 1$, tenemos

$$\begin{aligned} T(p) &= c_1 + T(p-1) + c_3 + c_4 + c_5 p + c_6 p - c_6 + c_7 p - c_7 + c_8 \\ &= T(p-1) + p(c_5 + c_6 + c_7) + (c_1 + c_3 + c_4 - c_6 - c_7) \\ &= T(p-1) + pk_1 + k_2 \end{aligned}$$

$$T(n) = \begin{cases} c_1 & n = 1 \\ T(n-1) + k_1 n + k_2 & \text{caso contrário} \end{cases}$$

Resolviendo la recurrencia:

$$
\begin{aligned}
T(n) &= T(n-1) + k_1 n + k_2 \\
&= T(n-2) + k_1(n-1) + k_2 + k_1 n + k_2 \\
&= T(n-2) + k_1(n-1+n) + 2k_2 \\
&= T(n-3) + k_1(n-2) + k_2 + k_1(n-1+n) + 2k_2 \\
&= T(n-3) + k_1(n-2+n-1+n) + 3k_2 \\
&= T(n-j) + k_1(n-j+1+\cdots+n-1+n) + jk_2 \\
&= T(1) + k_1(2+\cdots+n-1+n) + (n-1)k_2 \\
&= c_1 + k_1(\frac{n(n+1)}{2}) - 1) + (n-1)k_2 \\
&= \frac{k_1}{2}n^2 + (k_2 + \frac{k_1}{2})n + (c_1 - k_1 - k_2)
\end{aligned}
$$

Luego $T(n) = \Theta(n^2)$.

Ejercicio 3. Considere el siguiente problema de búsqueda. Entrada: un arreglo ordenado $A[1..n]$, y un número v. Salida: Un indice i tal que $v = A[i]$ si v está en A y -1 si v no está en A. El algoritmo de búsqueda binaria para dicho problema encuentra el punto medio de A y lo compara con v, descartando la mitad de la secuencia y repitiendo este procedimiento recursivamente. Escriba el pseudocódigo del algoritmo anterior. Escriba una recurrencia para el peor caso de este algoritmo. Resuelva la recurrencia.

Solucion

Recibe: un vector $A[p..q]$ de números enteros ordenado de manera creciente y un número entero v, y encuentra el índice del elemento v, y devuelve -1 si no existe dicho índice.

Busqueda-Binaria(A, p, q, v)	*cost*	*times*
if $p > q$	c_1	1
return -1	c_2	0
$r = \left\lfloor \frac{p+q}{2} \right\rfloor$	c_3	1
if $(A[r] == v)$	c_4	1
return r	c_5	0
if $(A[r] > v)$	c_6	1
return Busqueda-Binaria$(p, r-1)$	$T(r-p)$?	
return Busqueda-Binaria$(r+1, q)$	$T(q-r)$?	

Cuando $p > q$, en el peor caso, tendremos $T(q-p+1) = c_1 + 0c_2 + c_3 + c_4 + 0c_5 + c_6 + \max\{T(r-p), T(q-r)\} = d + T(\lfloor((q-p+1)/2)\rfloor)$

Luego

$$
T(n) = \begin{cases} c_1 & n = 1 \\ T(\left\lfloor \frac{n}{2} \right\rfloor) + d & \text{caso contrário} \end{cases}
$$

Resolveremos primero para $n = 2^k$ para algún k

$$
\begin{aligned}
T(n) &= T(2^k) \\
&= T(2^{k-1}) + d \\
&= T(2^{k-2}) + d + d \\
&= T(2^{k-j}) + dj \\
&= T(1) + dk \\
&= c_1 + dk \\
&= c_1 + d \lg n
\end{aligned}
$$

Ahora suponga que n no es potencia de 2. Luego $2^k \leq n < 2^{k+1}$ para algún entero k. Luego, como $T(n)$ es creciente (ver final),

$$T(n) < T(2^{k+1}) = c_1 + d(k+1) = c_1 + d + dk \leq c_1 + d + d \lg n$$

(la última desigualdad se obtiene porque $2^k \leq n$).

Vea también que

$$T(n) \geq T(2^k) = c_1 + dk = c_1 - d + d(k+1) > c_1 - d + d \lg n$$

(la última desigualdad se obtiene porque $2^{k+1} > n$).

Concluimos que

$$c - d + d \lg n < T(n) \leq c + d + d \lg n.$$

Portanto, $T(n) = \Theta(\lg n)$.

Faltaba demostrar que $T(n)$ es una función creciente. Demostraremos por inducción en n que $T(n) \leq T(n+1)$. Si $n = 1$, tenemos que $T(1) = 1 \leq 1 + d = T(1) + d = T(2)$. Si $n > 1$, tenemos dos casos.

Si n es par, $T(n) = T(\left\lfloor \frac{n}{2} \right\rfloor) + d = T(\left\lfloor \frac{n+1}{2} \right\rfloor) + d = T(n+1)$.

Si n es impar, $T(n) = T(\left\lfloor \frac{n}{2} \right\rfloor) + d \leq T(\left\lfloor \frac{n}{2} \right\rfloor + 1) + d = T(\left\lfloor \frac{n+1}{2} \right\rfloor) + d = T(n+1)$.

Ejercicio 4. Resolver las siguientes recurrencias. Compruebe usando inducción. Compruebe usando teorema maestro. En cada caso, suponga que $T(1) = 1$.

(a) $T(n) = 2T(\lfloor n/2 \rfloor) + n^2$

(b) $T(n) = 2T(n-1) + 3n - 2$

(c) $T(n) = 4T(\lfloor n/2 \rfloor) + n$

(d) $T(n) = 2T(\lfloor n/2 \rfloor) + n^3$

(e) $T(n) = 7T(\lfloor n/3 \rfloor) + n^2$

Solucion

(a) Resolveremos primero para $n = 2^k$ para algún k

$$
\begin{aligned}
T(n) &= T(2^k) \\
&= 2T(2^{k-1}) + (2^k)^2 \\
&= 2 \cdot (2T(2^{k-2}) + (2^{k-1})^2) + (2^k)^2 \\
&= 2^2T(2^{k-2}) + 2(2^{k-1})^2 + (2^k)^2 \\
&= 2^2(2T(2^{k-3}) + (2^{k-2})^2) + 2(2^{k-1})^2 + (2^k)^2 \\
&= 2^3T(2^{k-3}) + 2^2(2^{k-2})^2 + 2(2^{k-1})^2 + (2^k)^2 \\
&= 2^iT(2^{k-i}) + 2^{i-1}(2^{k-i+1})^2 + \cdots + (2^k)^2 \\
&= 2^kT(2^0) + 2^{k-1}(2^1)^2 + \cdots + (2^k)^2 \\
&= 2^kT(1) + 2^{k-1}(2^1)^2 + 2^{k-2}(2^2)^2 + \cdots + (2^k)^2 \\
&= 2^k + 2^{k-1}(2^1)^2 + 2^{k-2}(2^2)^2 + \cdots + (2^k)^2 \\
&= 2^k + \sum_{i=1}^{k} 2^{k-i}(2^i)^2 \\
&= 2^k + \sum_{i=1}^{k} 2^{k+i} \\
&= 2^k + 2^k \sum_{i=1}^{k} 2^i \\
&= 2^k \sum_{i=0}^{k} 2^i \\
&= 2^k(2^{k+1} - 1) \\
&= n(2n - 1)
\end{aligned}
$$

Vea que

Ahora suponga que n no es potencia de 2.

Luego $2^k \leq n < 2^{k+1}$ para algún entero k. Luego, como $T(n)$ es creciente (ejercicio),

$$T(n) < T(2^{k+1}) = 2^{k+1}(2(2^{k+1}) - 1) = 2 \cdot 2^k(4(2^k) - 1) \leq 2n(4n - 1)$$

(la última desigualdad se obtiene porque $2^k \leq n$).

Vea también que

$$T(n) \geq T(2^k) = 2^k(2^{k+1} - 1) = \frac{1}{2}2^{k+1}(2^{k+1} - 1) > \frac{n(n-1)}{2}$$

(la última desigualdad se obtiene porque $2^{k+1} > n$).

Concluimos que

$$\frac{n(n-1)}{2} < T(n) \leq 2n(4n-1).$$

Portanto, $T(n) = \Theta(n^2)$.
Falta: demostrar que $T(n)$ es una función creciente.

Comprobaremos por inducción que $T(n) = \Theta(n^2)$

- Primero probaremos por inducción en n que $T(n) \leq c_2 n^2$ para $n \geq 1$ para algunos $c_2 \geq 2, n_0 > 0$.Si $n = 1$, tenemos que $T(1) = 1 \leq 2 \leq c_2 n^2$. Si $n > 1$, tenemos que

$$\begin{aligned} T(n) &= 2T(\left\lfloor \frac{n}{2} \right\rfloor) + n^2 \\ &\leq 2c_2(\left\lfloor \frac{n}{2} \right\rfloor)^2 + n^2 \\ &\leq 2c_2(\frac{n}{2})^2 + n^2 \\ &= c_2 n^2(\frac{1}{2} + \frac{1}{c_2}) \\ &\leq c_2 n^2 \end{aligned}$$

- Ahora probaremos por inducción en n que $T(n) \geq c_1 n^2$ para $n \geq 1$ para algunos $c_2, n_0 > 0$ con $c_1 \leq 1$.Si $n = 1$, tenemos que $T(1) = 1 \geq c_1$. Si $n > 1$, tenemos que

$$\begin{aligned} T(n) &= 2T(\left\lfloor \frac{n}{2} \right\rfloor) + n^2 \\ &\geq 2T(\frac{n}{2} - 1) + n^2 \\ &\geq 2c_1(\frac{n}{2} - 1)^2 + n^2 \\ &\geq c_1 n^2(\frac{1}{2} - \frac{1}{n})^2 + n^2 \\ &= c_1 n^2((\frac{1}{2} - \frac{1}{n})^2 + \frac{1}{c_1}) \\ &\geq c_1 n^2(\frac{1}{c_1}) \\ &\geq c_1 n^2 \end{aligned}$$

Teorema maestro: Note que $a = 2$ $b = 2$ $k = 2$. y $\lg a / \lg b = \lg 2 / \lg 2 = 1 < 2$. Luego $T(n) = \Theta(n^k) = \Theta(n^2)$.

(d)

Resolveremos primero para $n = 2^k$ para algún k

$$
\begin{aligned}
T(n) &= T(2^k) \\
&= 2T(2^{k-1}) + (2^k)^3 \\
&= 2 \cdot (2T(2^{k-2}) + (2^{k-1})^3) + (2^k)^3 \\
&= 2^2T(2^{k-2}) + 2(2^{k-1})^3 + (2^k)^3 \\
&= 2^2(2T(2^{k-3}) + (2^{k-2})^3) + 2(2^{k-1})^3 + (2^k)^3 \\
&= 2^3T(2^{k-3}) + 2^2(2^{k-2})^3 + 2(2^{k-1})^3 + (2^k)^3 \\
&= 2^iT(2^{k-i}) + 2^{i-1}(2^{k-i+1})^3 + \cdots + (2^k)^3 \\
&= 2^kT(1) + 2^{k-1}(2^1)^3 + 2^{k-2}(2^2)^3 + \cdots + (2^k)^3 \\
&= 2^k + 2^{k-1}(2^1)^3 + 2^{k-2}(2^2)^3 + \cdots + (2^k)^3 \\
&= 2^k(2^0)^3 + 2^{k-1}(2^1)^3 + 2^{k-2}(2^2)^3 + \cdots + (2^k)^3 \\
&= \sum_{i=0}^{k} 2^{k-i}(2^i)^3 \\
&= \sum_{i=0}^{k} 2^{k+2i} \\
&= 2^k \sum_{i=0}^{k} 2^{2i} \\
&= 2^k \sum_{i=0}^{k} 4^i \\
&= 2^k(4^{k+1} - 1)/3 \\
&= n(4n^2 - 1)/3
\end{aligned}
$$

Ahora suponga que n no es potencia de 2.

Luego $2^k \leq n < 2^{k+1}$ para algún entero k. Luego, como $T(n)$ es creciente (ejercicio),

$$T(n) < T(2^{k+1}) = \frac{1}{3}2^k(4^{k+1} - 1) = \frac{1}{3}2^{k+1}(4^{k+2} - 1) \leq \frac{2n}{3}(16n^2 - 1)/3$$

(la última desigualdad se obtiene porque $2^k \leq n$).

Vea también que

$$T(n) \geq T(2^k) = \frac{1}{3}2^k(4^{k+1} - 1) = \frac{1}{6}2^{k+1}(4^{k+1} - 1) > \frac{1}{6}n(n^2 - 1)$$

(la última desigualdad se obtiene porque $2^{k+1} > n$).

Concluimos que

$$\frac{n(n^2 - 1)}{6} < T(n) \leq \frac{2n}{9}(16n^2 - 1).$$

Portanto, $T(n) = \Theta(n^3)$.
Falta: demostrar que $T(n)$ es una función creciente.

Comprobaremos por inducción que $T(n) = \Theta(n^3)$

- Primero probaremos por inducción en n que $T(n) \leq c_2 n^3$ para $n \geq 1$ para algunos $c_2 \geq 4/3, n_0 > 0$. Si $n = 1$, tenemos que $T(1) = 1 \leq 2 \leq c_2 n^3$. Si $n > 1$, tenemos que

$$\begin{aligned}
T(n) &= 2T(\left\lfloor \frac{n}{2} \right\rfloor) + n^3 \\
&\leq 2c_2(\left\lfloor \frac{n}{2} \right\rfloor)^3 + n^3 \\
&\leq 2c_2(\frac{n}{2})^3 + n^3 \\
&= c_2 n^3(\frac{1}{4} + \frac{1}{c_2}) \\
&\leq c_2 n^3
\end{aligned}$$

- Ahora probaremos por inducción en n que $T(n) \geq c_1 n^3$ para $n \geq 1$ para algunos $c_2, n_0 > 0$ con $c_1 \leq 1$.Si $n = 1$, tenemos que $T(1) = 1 \geq c_1$. Si $n > 1$, tenemos que

$$\begin{aligned}
T(n) &= 2T(\left\lfloor \frac{n}{2} \right\rfloor) + n^3 \\
&\geq 2T(\frac{n}{2} - 1) + n^3 \\
&\geq 2c_1(\frac{n}{2} - 1)^3 + n^3 \\
&\geq c_1 n^3(\frac{1}{2} - \frac{1}{n})^3 + n^3 \\
&= c_1 n^3((\frac{1}{2} - \frac{1}{n})^3 + \frac{1}{c_1}) \\
&\geq c_1 n^3(\frac{1}{c_1}) \\
&\geq c_1 n^3
\end{aligned}$$

Teorema maestro: Note que $a = 2$ $b = 2$ $k = 2$. y $\lg a / \lg b = \lg 2 / \lg 2 = 1 < 2$. Luego $T(n) = \Theta(n^k) = \Theta(n^2)$.