Analisis y Diseño de Algoritmos

Juan Gutiérrez

April 21, 2022

Subarreglo máximo

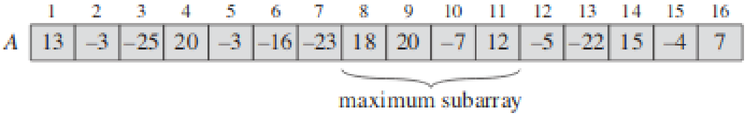

Figure: Tomada del libro Cormen, Introduction to Algorithms.

Subarreglo máximo

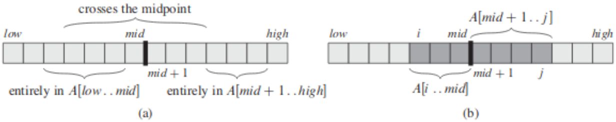

Figure: Tomada del libro Cormen, Introduction to Algorithms

Subarreglo máximo


```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
left-sum = -∞
sum = 0
for i = mid downto low
    sum = sum + A[i]
    if sum > left-sum
        left-sum = sum
        max-left = i
right-sum = -∞
sum = 0
for j = mid + 1 to high
    sum = sum + A[j]
    if sum > right-sum
        right-sum = sum
        max-right = j
return (max-left, max-right, left-sum + right-sum)
```

Subarreglo máximo


```
FIND-MAXIMUM-SUBARRAY(A, low, high)
if high == low
    return (low, high, A[low])            // base case: only one element
else mid = ⌊(low + high)/2⌋
    (left-low, left-high, left-sum) =
        FIND-MAXIMUM-SUBARRAY(A, low, mid)
    (right-low, right-high, right-sum) =
        FIND-MAXIMUM-SUBARRAY(A, mid + 1, high)
    (cross-low, cross-high, cross-sum) =
        FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
    if left-sum ≥ right-sum and left-sum ≥ cross-sum
        return (left-low, left-high, left-sum)
    elseif right-sum ≥ left-sum and right-sum ≥ cross-sum
        return (right-low, right-high, right-sum)
    else return (cross-low, cross-high, cross-sum)
```

Figure: Tomada del libro Cormen, Introduction to Algorithms

Multiplicación de números naturales

9999	A
7777	B
69993	C
69993	D
69993	E
69993	F
77762223	G

Multiplicación de números naturales

Recibe: Dos números enteros representados por $a[1..n], b[1..n]$
Devuelve: el producto $a \cdot b$

Multiplicacion-basica(a, b, n)	*cost*	*times*
total $= 0$	c_1	1
for $j = 1$ **to** n	c_2	$n + 1$
sum $= 0$	c_3	n
for $i = 1$ **to** n	c_4	$(n+1) \cdot$
sum $=$ sum $\cdot 10 + b[j] \cdot a[i]$	c_5	$n \cdot n$
total $=$ total $\cdot 10 +$ sum	c_6	n
return total	c_7	1

Multiplicación de números naturales

Require: Dos números enteros a y b de n dígitos, donde n es una potencia de dos, y tanto a como b no contienen ceros.

Ensure: El producto $a \cdot b$

MULTIPLICACION-DC (a, b, n)	*cost*	*times*
if $n = 1$	$\Theta(1)$	1
return $a \cdot b$	$\Theta(n)$	1
$a_1 = \lfloor a/10^{n/2} \rfloor$	$\Theta(n)$	1
$a_2 = a \mod 10^{n/2}$	$\Theta(n)$	1
$b_1 = \lfloor b/10^{n/2} \rfloor$	$\Theta(n)$	1
$b_2 = b \mod 10^{n/2}$	$\Theta(n)$	1
$p = \text{MULTIPLICACION-DC}(a_1, b_1, n/2)$	$T(n/2)$	1
$q = \text{MULTIPLICACION-DC}(a_1, b_2, n/2)$	$T(n/2)$	1
$r = \text{MULTIPLICACION-DC}(a_2, b_1, n/2)$	$T(n/2)$	1
$s = \text{MULTIPLICACION-DC}(a_2, b_2, n/2)$	$T(n/2)$	1
return $p \cdot 10^n + (q + r) \cdot 10^{n/2} + s$	$\Theta(n)$	1

Multiplicación de números naturales

Require: Dos números enteros a y b de n dígitos, donde n es una potencia de 2 y tanto a como b no contienen ceros

Ensure: El producto $a \cdot b$

KARATSUBA (a, b)	*cost*	*times*
if $n \leq 1$	$\Theta(1)$	1
return $a \cdot b$	$\Theta(n)$	1
$a_1 = \lfloor a/10^{n/2} \rfloor$	$\Theta(n)$	1
$a_2 = a \mod 10^{n/2}$	$\Theta(n)$	1
$b_1 = \lfloor b/10^{n/2} \rfloor$	$\Theta(n)$	1
$b_2 = b \mod 10^{n/2}$	$\Theta(n)$	1
$p = \text{KARATSUBA}(a_1, b_1)$	$T(n/2)$	1
$q = \text{KARATSUBA}(a_1 + a_2, b_1 + b_2)$	$T(n/2)$	1
$s = \text{KARATSUBA}(a_2, b_2)$	$T(n/2)$	1
return $p \cdot 10^n + (q - p - s) \cdot 10^n + s$	$\Theta(n)$	1

Multiplicación de matrices

SQUARE-MATRIX-MULTIPLY(A, B)

```
n = A.rows
let C be a new n × n matrix
for i = 1 to n
    for j = 1 to n
        c_ij = 0
        for k = 1 to n
            c_ij = c_ij + a_ik · b_kj
return C
```

Multiplicación de matrices

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}, \quad C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix},$$

Multiplicación de matrices

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}.$$

Multiplicación de matrices

$$
\begin{aligned}
C_{11} &= A_{11} \cdot B_{11} + A_{12} \cdot B_{21} \,, \\
C_{12} &= A_{11} \cdot B_{12} + A_{12} \cdot B_{22} \,, \\
C_{21} &= A_{21} \cdot B_{11} + A_{22} \cdot B_{21} \,, \\
C_{22} &= A_{21} \cdot B_{12} + A_{22} \cdot B_{22} \,.
\end{aligned}
$$

Multiplicación de matrices

SQUARE-MATRIX-MULTIPLY-RECURSIVE(A, B)

$n = A.rows$
let C be a new $n \times n$ matrix
if $n == 1$
    $c_{11} = a_{11} \cdot b_{11}$
else partition A, B, and C as in equations (4.9)
    $C_{11} =$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{11}, B_{11})
        $+$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{12}, B_{21})
    $C_{12} =$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{11}, B_{12})
        $+$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{12}, B_{22})
    $C_{21} =$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{21}, B_{11})
        $+$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{22}, B_{21})
    $C_{22} =$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{21}, B_{12})
        $+$ SQUARE-MATRIX-MULTIPLY-RECURSIVE(A_{22}, B_{22})
return C

Multiplicación de matrices

1. Divide the input matrices A and B and output matrix C into $n/2 \times n/2$ submatrices, as in equation (4.9). This step takes $\Theta(1)$ time by index calculation, just as in SQUARE-MATRIX-MULTIPLY-RECURSIVE.
2. Create 10 matrices $S_1, S_2, \ldots, S_{10}$, each of which is $n/2 \times n/2$ and is the sum or difference of two matrices created in step 1. We can create all 10 matrices in $\Theta(n^2)$ time.
3. Using the submatrices created in step 1 and the 10 matrices created in step 2, recursively compute seven matrix products $P_1, P_2, \ldots, P_7$. Each matrix P_i is $n/2 \times n/2$.
4. Compute the desired submatrices $C_{11}, C_{12}, C_{21}, C_{22}$ of the result matrix C by adding and subtracting various combinations of the P_i matrices. We can compute all four submatrices in $\Theta(n^2)$ time.

Multiplicación de matrices

$$
\begin{aligned}
S_1 &= B_{12} - B_{22} ,\\
S_2 &= A_{11} + A_{12} ,\\
S_3 &= A_{21} + A_{22} ,\\
S_4 &= B_{21} - B_{11} ,\\
S_5 &= A_{11} + A_{22} ,\\
S_6 &= B_{11} + B_{22} ,\\
S_7 &= A_{12} - A_{22} ,\\
S_8 &= B_{21} + B_{22} ,\\
S_9 &= A_{11} - A_{21} ,\\
S_{10} &= B_{11} + B_{12} .
\end{aligned}
$$

Multiplicación de matrices

$$
\begin{aligned}
P_1 &= A_{11} \cdot S_1 &&= A_{11} \cdot B_{12} - A_{11} \cdot B_{22} \,, \\
P_2 &= S_2 \cdot B_{22} &&= A_{11} \cdot B_{22} + A_{12} \cdot B_{22} \,, \\
P_3 &= S_3 \cdot B_{11} &&= A_{21} \cdot B_{11} + A_{22} \cdot B_{11} \,, \\
P_4 &= A_{22} \cdot S_4 &&= A_{22} \cdot B_{21} - A_{22} \cdot B_{11} \,, \\
P_5 &= S_5 \cdot S_6 &&= A_{11} \cdot B_{11} + A_{11} \cdot B_{22} + A_{22} \cdot B_{11} + A_{22} \cdot B_{22} \,, \\
P_6 &= S_7 \cdot S_8 &&= A_{12} \cdot B_{21} + A_{12} \cdot B_{22} - A_{22} \cdot B_{21} - A_{22} \cdot B_{22} \,, \\
P_7 &= S_9 \cdot S_{10} &&= A_{11} \cdot B_{11} + A_{11} \cdot B_{12} - A_{21} \cdot B_{11} - A_{21} \cdot B_{12} \,.
\end{aligned}
$$

Multiplicación de matrices

Recibe: Dos matrices A y B de dimensiones $n \times n$ Devuelve: $A \cdot B$

Strassen (A, B) *cost* *times*

if $n = 1$

  $c_{11} = a_{11} \cdot b_{11}$

else

  Crear las matrices auxiliares

  P_1=Multiplica-DC(A_{11}, S_1)

  P_2=Multiplica-DC(S_2, B_{22})

  P_3=Multiplica-DC(S_3, B_{11})

  P_4=Multiplica-DC(A_{22}, S_4)

  P_5=Multiplica-DC(S_5, S_6)

  P_6=Multiplica-DC(S_7, S_8)

  P_7=Multiplica-DC(S_9, S_{10})

  C_{11}=$P_5 + P_4 - P_2 + P_6$

  C_{12}=$P_1 + P_2$

Inversiones

Ejemplo, sea $A = [2, 4, 1, 3, 5]$. Las inversiones son $(1, 3), (2, 3), (2, 4)$

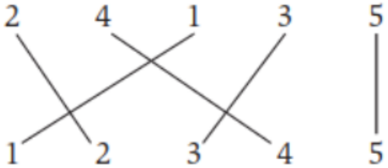

Figure 11: Tomada del libro Kleinberg-Tardos, Algorithm Design

Inversiones

Recibe: Un vector de números enteros diferentes $A[1..n]$
Devuelve: El número de inversiones en A.

INVERSIONES-INGENUO(A, n)

```
total = 0
for i = 1 to n − 1
    for j = i + 1 to n
        if A[i] > A[j]
            total = total + 1
return total
```

Inversiones

Si (i, j) es una inversión con $i \in \{1, \ldots, \lfloor n/2 \rfloor\}$ y $j \in \{\lfloor n/2 \rfloor + 1, \ldots, n\}$ entonces (i', j) también es una inversión para todo $i' > i$

Inversiones

Recibe: Un vector de números enteros diferentes $A[1..n]$ y tres índices p, q, r tales que $A[p..q]$ y $A[q+1..r]$ están ordenados.

Devuelve: El número de inversiones (i, j) en A tales que $i \in \{p, \ldots, q\}$, $j \in \{q+1, \ldots, r\}$. Además, ordena el vector $A[p..r]$.

Inversiones-Centradas(A, p, q, r)

$n_1 = q - p + 1$
$n_2 = r - q$
Let $L[1..n_1 + 1]$ and $R[1..n_2 + 1]$ be new arrays
for $i = 1$ **to** n_1
    $L[i] = A[p + i - 1]$
for $j = 1$ **to** n_2
    $R[j] = A[q + j]$
$L[n_1 + 1] = \infty$
$L[n_2 + 1] = \infty$
$i = 1$
$j = 1$
$total = 0$
for $k = p$ **to** r
    if $L[i] \leq R[j]$
        $A[k] = L[i]$
        $i = i + 1$
        $total = total + j - 1$
    else
        $A[k] = R[j]$
        $j = j + 1$
return total

Inversiones

Recibe: Un vector de números enteros diferentes $A[p..r]$
Devuelve: El número de inversiones en A.

INVERSIONES-DC(A, p, r)	*cost*	*times*
if $(p == r)$	c_1	*1*
return 0	c_2	*0*
$q = \left\lfloor \frac{r-p+1}{2} \right\rfloor$	c_3	1
$total_1$ = INVERSIONES-DC(A, p, q)	$T(\lfloor n/2 \rfloor)$	1
$total_2$ = INVERSIONES-DC$(A, q+1, r)$	$T(\lceil n/2 \rceil)$	1
$total_3$ = INVERSIONES-CENTRADAS(A, p, q, r)	kn	1
return $total_1 + total_2 + total_3$	c_5	1

Gracias