Algoritmo Karatsuba para la multiplicación de enteros grandes

Informe - Diseño y Análisis de Algoritmos

Grupo de Diseño y Análisis de Algoritmos L2_5

Ernesto Echeverría González
Germán Pescador Barreto
Sebastián José Díaz Rodríguez

Índice

1.- Introducción

Anatoly Alexeevitch Karatsuba, nacido en 1937 en Grozni (por aquel entonces ciudad de la Unión Soviética y actualmente capital de la República de Chechenia en Rusia) y fallecido en 2008 en Moscú, diseñó el conocido como algoritmo de Karatsuba en el año 1960. Tras el intento de Andrey Kolmogorov de probar en un seminario de problemas matemáticos cibernéticos que no había algoritmo capaz de multiplicar números grandes en menos de $\Omega(n^2)$ operaciones elementales; Karatsuba, en ese momento un estudiante de 23 años, halló un algoritmo divide y vencerás que multiplica dos números de longitud n en $\theta(n^{log_2 3})$, refutando así la suposición inicial de Kolmogorov. Dos años después, el algoritmo fue publicado en la revista *Proceedings of the USSR Academy of Sciences,* estableciéndose como el primer algoritmo divide y vencerás de la historia.

El algoritmo Karatsuba fue el primer algoritmo de multiplicación asintóticamente más rápido que el método cuadrático, aunque en la actualidad existen algoritmos que lo superan en velocidad para cantidades de n elementos lo suficientemente grandes. Un ejemplo es el algoritmo Toom-Cook, que reduce la complejidad a $\theta(n^{log(5)/log(3)})$.

2.- Descripción del problema

El algoritmo Karatsuba es un algoritmo de multiplicación rápida utilizado para el producto de dos enteros de n dígitos. Para ello supone dos números x e y como:

$$x = x_1 B^m + x_0$$
$$y = y_1 B^m + y_0$$

Donde B es la base de la multiplicación (generalmente decimal, es decir, 10) y m un número menor que el número de dígitos n, aunque generalmente se escoge la mitad de n. Por ejemplo, si x fuera 12345, se podría expresar como $12 \cdot 10^3 + 345$.

$$xy = (x_1 B^m + x_0)(y_1 B^m + y_0) = z_2 B^{2m} + z_1 B^m + z_0$$
$$z_2 = x_1 y_1$$
$$z_1 = x_1 y_0 + x_0 y_1 = (x_1 + x_0)(y_1 + y_0) - z_2 - z_0$$
$$z_0 = x_0 y_0$$

3.- Pseudocódigo y análisis de complejidad

Pseudocódigo del algoritmo Karatsuba para multiplicación de enteros

```
begin Karatsuba(num1, num2)
    if (num1 < 10) or (num2 < 10)
        return [ num1*num2 ]

    /* Cálculo del número de dígitos de los subproblemas */
    m = max( len_num1, len_num2 ) / 2

    /* División de las cifras en subcifras para la resolución del problema */
    high1 = upper_half(num1, m)
    low1 = lower_half(num1, m)

    high2 = upper_half(num2, m)
    low2 = lower_half(num2, m)

    /* Llamadas recursivas para la combinación de las subsoluciones */
    z0 = karatsuba(low1,low2)
    z1 = karatsuba( (low1+high1) , (low2+high2) )
    z2 = karatsuba(high1,high2)

    return [ z2*10^(2*m) + (z1-z2-z0)*10^m + z0 ]
end Karatsuba
```

Tomando el trabajo del algoritmo Karatsuba como T(n) = 3T(n/2), debido a las tres llamadas recursivas para la resolución de los subproblemas de una longitud igual a la mitad del que parten, se puede deducir a través del método maestro una complejidad $\theta(n^{log_2 3})$. El cálculo de complejidad mediante el método maestro es posible gracias a que los subproblemas en los que se divide son de longitud igual.

Partiendo de T(n) ≤ aT(n/b) + O(n^d) y del trabajo realizado por el algoritmo, T(n) = 3T(n/2), se obtiene el caso 3 del método maestro, O(n^$\log_b$a).

a = 3 b = 2 d = 1
a > b^d
O(n^$\log_b$a) = O(n^$\log_2$3)

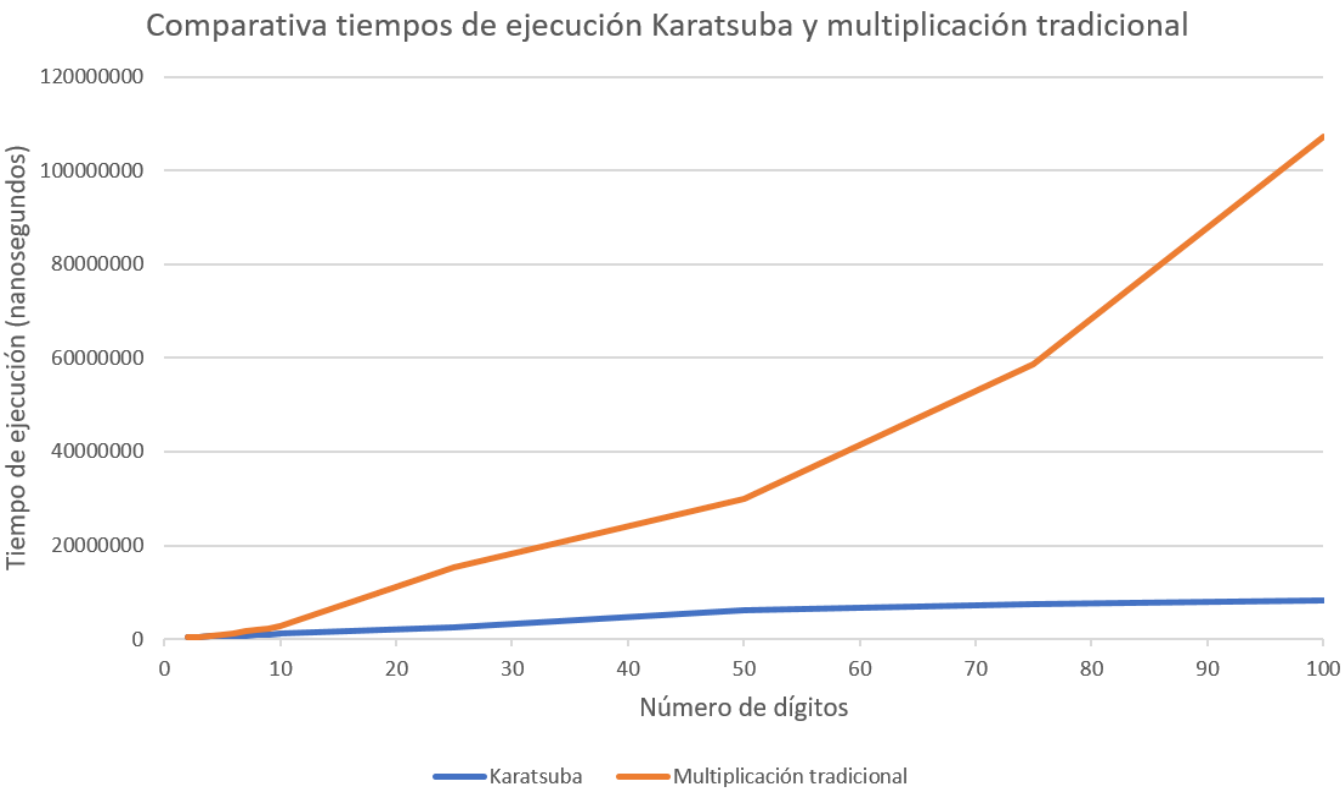
Esta cota superior nos permite comparar su complejidad frente al algoritmo tradicional. En el peor de los casos el algoritmo Karatsuba seguirá siendo más eficiente que el algoritmo tradicional.

Pseudocódigo del algoritmo tradicional para multiplicación de enteros

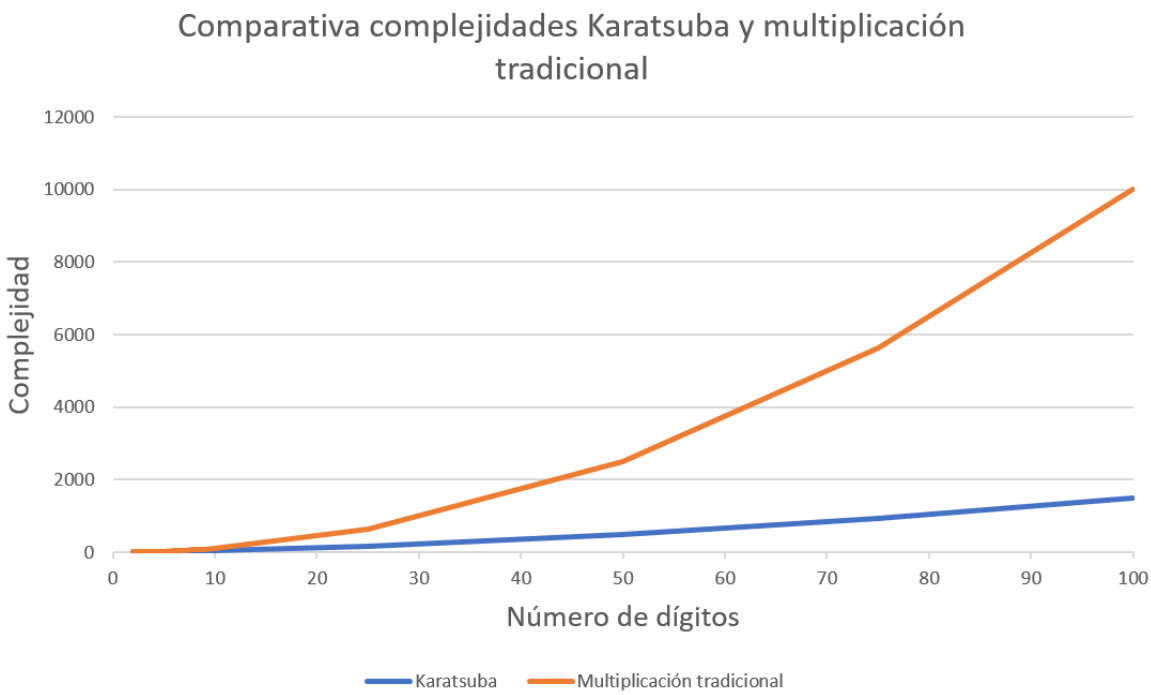
```
begin Tradicional(num1, num2)
        for i = 0 to len_num2
                for j = 0 to len_num1
                        resultado += num1[ j ] * num2[ i ] * 10^(i+j)
                for end
        for end
        return resultado
end Tradicional
```

La complejidad de este algoritmo es a simple vista deducible. Debido a los bucles de longitud máxima n, su complejidad es $\theta(n^2)$, tomando la operación interna como O(1).

4.- Evaluación experimental



Número de dígitos	Tiempo Karatsuba	Complejidad Karatsuba	Tiempo Tradicional	Complejidad Tradicional
2	471828 ns	3,000077979	346249 ns	4
3	495073 ns	5,70475751	412963 ns	9
4	559673 ns	9,000467878	532806 ns	16
5	559975 ns	12,81939286	830152 ns	25
6	668348 ns	17,11471738	1247039 ns	36
7	774909 ns	21,85145667	1727018 ns	49
8	800568 ns	27,00210548	1949498 ns	64
9	994974 ns	32,54425824	2267372 ns	81
10	1144402 ns	38,4591782	2819196 ns	100
25	2388422 ns	164,3368332	15378035 ns	625
50	6212857 ns	493,0233143	29976934 ns	2500
75	7454160 ns	937,5017832	58604795 ns	5625
100	8312990 ns	1479,108388	107269504 ns	10000



5.- Conclusiones

Con el estudio exhaustivo del algoritmo Karatsuba y su comparación con el algoritmo tradicional queda demostrada la capacidad de los algoritmos basados en la metodología Divide y Vencerás para obtener complejidades inferiores a la cuadrática.

A partir de un N lo suficientemente grande, en el caso experimental aplicado a partir de un N igual a cinco, el algoritmo de Karatsuba es capaz de obtener mejores resultados en cuanto a rendimiento.

Cabe resaltar dentro de los resultados del estudio que la capacidad del algoritmo Karatsuba se ha visto desafiada por nuevos algoritmos a lo largo de los años. Ya habíamos comentado el algoritmo Toom-Cook, que obtiene resultados mucho mejores que el algoritmo Karatsuba al multiplicar números extremadamente grandes. Un resultado de esta búsqueda de eficiencia se puede observar en el lenguaje Java, mediante el cual se programaron los algoritmos del estudio, donde internamente se aplican diferentes algoritmos para las estructuras BigInteger según la cantidad de dígitos que contengan sus números.

6.- Bibliografía

Explicación inical del algoritmo Karatsuba

Biografía de Anatoly Karatsuba

Comparación de complejidad entre algoritmos tradicional, Karatsuba y Toom-Cook

Explicación mediante traza del algoritmo Karatsuba

Complejidad del algoritmo Karatsuba