ALGORITMO DE KARATSUBA EN CONTEXTO ADITIVO

KARATSUBA ALGORITHM IN ADDITIVE CONTEXT

Jesús Ayuso Pérez
Compositor musical. Licenciado en Ingeniería Informática por la Universidad Carlos III de Madrid (UC3M) (España).
E-mail: ayusoperez@terra.com ORCID: https://orcid.org/0000-0001-8287-6089

Recepción: 06/04/2018. **Aceptación:** 15/06/2018. **Publicación:** 28/09/2018

Citación sugerida:
Ayuso Pérez, J. (2018). Algoritmo de Karatsuba en contexto aditivo. *3C TIC. Cuadernos de desarrollo aplicados a las TIC*, 7(3), 10-21. doi:http://dx.doi.org/10.17993/3ctic.2018.61.10-21/

RESUMEN

El algoritmo dado por Anatoly Alexeevitch Karatsuba en 1960 (Karatsuba, 1962) para la multiplicación no es únicamente aplicable en un contexto multiplicativo, se puede aplicar a cualquier contexto algebraico que se defina partiendo de una relación de equivalencia bien formada previa (Ayuso 2018). El hecho de que el citado método sea perfectamente extrapolable a distintos ámbitos algebraicos (Ayuso, 2013-2018) abre la puerta a su utilización en un contexto aditivo. De ahí que en el presente documento se proponga un algoritmo de adición entre enteros basado en dicho concepto.

ABSTRACT

The algorithm given by Anatoly Alexeevitch Karatsuba in 1960 (Karatsuba, 1962) for multiplication does not apply only in a multiplicative context, it can be applied to an algebraic context that is defined starting from a previous well-formed equivalence relation (Ayuso 2018). The fact that the aforementioned method is perfectly extrapolated to different algebraic areas (Ayuso, 2013-2018) opens the door to its use in an additive context. Hence, in this document, an integer selection algorithm based on this concept was proposed.

PALABRAS CLAVE

Karatsuba, Algoritmo, Adición, Sucesor, Grupo Abeliano.

KEY WORDS

Karatsuba, Algorithm, Addition, Successor, Abelian group.

Ed. 26. Vol.7 Nº 3. Septiembre-Diciembre 2018
DOI: http://dx.doi.org/10.17993/3ctic.2018.61.10-21/

1. INTRODUCCIÓN

De la misma manera que la multiplicación entre enteros de longitud *n* puede descomponerse (Karatsuba, 1962) siguiendo un criterio de partición binaria, la adición entre enteros, por ejemplo *u* más *v*, es igualmente afrontable a través del mismo tipo de subdivisión:

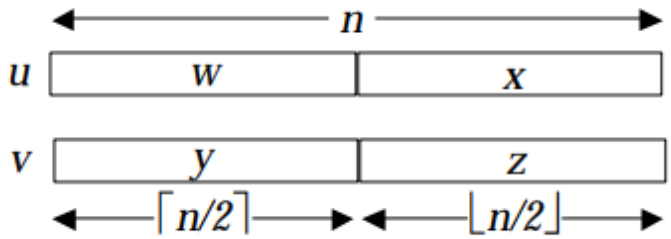

Ilustración 1. Ilustración de la descomposición de Karatsuba.
Fuente: elaboración propia.

Con el apoyo de la bibliografía, se puede recordar cómo dicha operación dentro de un contexto multiplicativo, *u* * *v*, se expresa matemáticamente usando el citado algoritmo como:

$$u * v = (w\ y)\ 10^{n} + (w\ z + x\ y)\ 10^{n/2} + x\ z$$

Fórmula 1. Algoritmo de Karatsuba de multiplicación entre enteros.

Con la definición en Fórmula 1, y teniendo en cuenta que la adición entre enteros de longitud *n* puede definirse como una composición de cálculos de sucesores (Booth, 1951), de manera que se denota la operación de cálculo del *y-esimo* sucesor de *w* como:

$$s^{0}(w) = w$$

$$s^{1}(w) = \text{successor}(w)$$

$$s^{y}(w) = s(s(\ .\ .\ .\ s(w)\ .\ .\ .\ ))$$

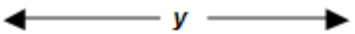

Ilustración 2. Notación para operación de cálculos de sucesores.

Partiendo de la nomenclatura anterior, sin requerir de más desarrollo matemático, el resultado directo de aplicar la expresión que figura en la Fórmula 1 al cómputo de operaciones en nuestro contexto aditivo quedaría modelado con la fórmula:

$$u+v = (s^1(0) \vee s^0(0))\,10^n \mid (s^1(w+y) \vee s^0(w+y))\,10^{n/2} \mid (x+z)$$

Fórmula 2. Algoritmo de Karatsuba de adición entre enteros.

Dicho esto, se entenderá la adición de un entero como la concatenación de elementos sucesores de los elementos involucrados en la operación teniendo en cuenta ciertas potencias por la base en la que se está trabajando, en este caso: base 10; las cuales suelen significar simples desplazamientos. Entendidos estos últimos como una acción de trasladar dígitos, *d*, en determinada base, *B*, cambiando su peso significativo; tal que como refleja la Ilustración 3:

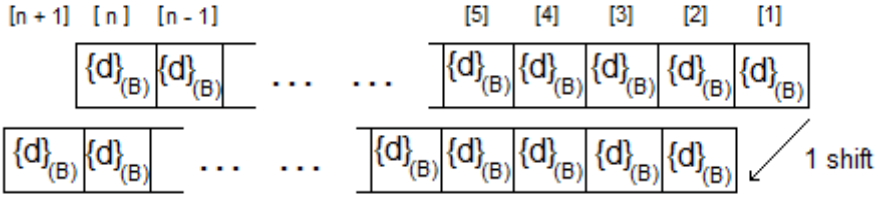

Ilustración 3. Técnica de desplazamiento de dígitos en determinada Base.

Fuente: elaboración propia.

2. METODOLOGÍA

Asentando un marco teórico inicial más específico que servirá de apoyo más detallado, y partiendo de que en la presente exposición se hará uso exclusivamente de pseodocódigo, en el sentido más estrictamente computacional, para exponer los distintos algoritmos y métodos propuestos, sin vincularlos a ningún lenguaje de programación concreto, pues el análisis presentado en este artículo

> Se entenderá la adición de un entero como la concatenación de elementos sucesores de los elementos involucrados en la operación teniendo en cuenta ciertas potencias por la base en la que se está trabajando, en este caso: base 10.

El método de optimización original fue concebido por Karatsuba dentro de un contexto multiplicativo, consiguiendo demostrar que el producto de 2 enteros podía calcularse en un orden de complejidad inferior al que estaba tradicionalmente aceptado: θ(n^2). De ahí el más que justificado interés por tratar de rebajar ese θ(n) que se presupone como orden de complejidad para la suma de 2 enteros.

DOI: http://dx.doi.org/10.17993/3ctic.2018.61.10-21/

El concepto que explotó Karatsuba fue el de hacer un particionado, el cual se suele modelar de una manera muy limpia a través de una estrategia recursiva. Dicha fragmentación produce la ventaja de que los cálculos parciales de esa partición son reutilizables para computar de una forma más ligera el cálculo más general, en que se subdivide cada tarea, de la deconstrucción de la operación multiplicativa (Ayuso, 2013-2018). Con ellos, dentro del contexto multiplicativo en el que, como se ha indicado, Karatsuba definió su algoritmo, lograba ahorrar una multiplicación de la susodicha subdivisión, en ese proceso de fragmentación, consiguiendo el mismo efecto algebraico a través de operaciones aditivas: más concretamente, sumas y restas, las cuales son computacionalmente menos costosas; similar al concepto empleado por Booth (Booth, 1951) pero sin esa partición binaria.
Existen alternativas documentadas o mejoras para el algoritmo primigenio descrito por Karatsuba, por ejemplo, emplearlo sobre cuerpos de otras características: generalizándolo para la multiplicación entre polinomios, o soluciones que explotan su susceptibilidad de ser paralelizado… El presente artículo se desvincula absolutamente de esas líneas de investigación, haciendo un cambio radical del contexto algebraico en que el Karatsuba ideó su método.

Al margen de los estudios derivados de este concepto, el proceso mostrado aquí será igualmente dividir el cálculo total en subproblemas más simples y, del mismo modo que, para el método original, asustentar en la menor complejidad computación de las operaciones que componen a aquella que se está calculando, evidentemente manteniendo la validez algebraica de las transformaciones, para que el resultado sea el correcto, tal y como hizo Karatsuba (1962). Con la consiguiente aportación que esto supone, es decir, dando una mayor magnitud a la mentada herramienta algorítmica, y la aceleración de la operación aditiva que nos ocupa.

```
result = m;

for(int i = 0; i < n; i++)
      result = successor(result);
```

Algoritmo 1. Algoritmo para el cálculo del sucesor (successor).

Con la anterior operación mencionada en Algoritmo 1 ya se puede describir el algoritmo de adición mediante la técnica descrita por Karatsuba para la multiplicación. Aplicando la fórmula que consta en Fórmula 2 se obtendría que el resultado de sumar a un entero u de longitud n el entero v también de longitud n sería:

```
if(n == 1)    // CASO BASE
     return successor(v, u);

w = u / 10^(n / 2);
x = u % 10^(n / 2);

y = v / 10^(n / 2);
z = v % 10^(n / 2);

t1 = addKaratsuba(x, z);

if( length(t1) > n / 2 )                // desplz. Base 10
   return successor(1, addKaratsuba(w, y)) * 10^(n / 2)
        + (t1 % 10^(n / 2));

t2 = addKaratsuba(w, y);

if( length(t2) > n / 2 )
   return successor(1, 0) * 10^n        // desplz. Base 10
        + (t2 % 10^(n / 2)) * 10^(n / 2) + t1;

     return t2 * 10^(n / 2) + t1;
```

Algoritmo 2. Algoritmo recursivo de adición por Karatsuba (addKaratsuba).

Se entiende que la operación length del código que figura sobre estas líneas, devuelve el número de dígitos, en base 10 en este caso, de los que consta el operando.

En primer lugar, puntualizar las operaciones de módulos, divisiones y productos por la base en el Algoritmo 2. Han sido expresados de esa manera en el anterior algoritmo para mejorar la legibilidad de los mismos, pero a efectos prácticos se pueden traducir en la obtención de los n dígitos superiores o inferiores de determinado operando y desplazamientos, algo que se destaca porque está relacionado con la eficiencia que puede esperarse de la presente solución. En segundo lugar, hacer notar como igualmente las operaciones de concatenación de valores han sido expresadas

Ed. 26. Vol.7 Nº 3. Septiembre-Diciembre 2018
DOI: http://dx.doi.org/10.17993/3ctic.2018.61.10-21/

mediante el símbolo '+' y ciertos desplazamientos supeditados a la base en la que se está trabajando. Ídem, su implementación real podría abordarse de una manera mucho más óptima, como simples concatenaciones y establecimientos.

Para conseguir visualizar mejor la descomposición binaria que se hace del problema mediante la técnica de Karatsuba, se utilizarán varios ejemplos y se intentará dar una representación gráfica del particionado en subproblemas que se realiza de la operación original.

En el primero de ellos, se va a trabajar en una nomenclatura diferente de la base decimal para poder profundizar en el concepto de elemento sucesor del contexto algebraico, es por ello que se utilizará la siguiente tabla de símbolos:

Nº decimal	símbolo	Nº decimal	símbolo	Nº decimal	símbolo
0	0	7	7	14	E
1	1	8	8	15	F
2	2	9	9	16	G
3	3	10	A	17	H
4	4	11	B	18	I
5	5	12	C	19	J
6	6	13	D		

Ilustración 4. Tabla de acciones de Booth.
Fuente: elaboración propia.

Establecida la anterior tabla de símbolos, se muestra el primero ejemplo, en el que se van a sumar los números enteros: 2354 + 9768`.

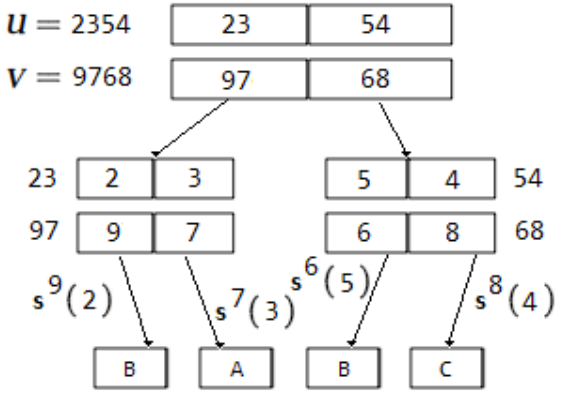

Ilustración 5. Técnica de particionado binario mediante Karatsuba, e.g. 2354 + 9768.
Fuente: elaboración propia.

La figura anterior ilustra la descomposición recursiva realizada por el algoritmo de Karatsuba aplicada a la relación aditiva definida sobre los números enteros. Se observa cómo el cómputo final consiste únicamente en ubicar en la posición correspondiente el elemento en cuestión, que es el resultado de moverse ordinalmente, desde un elemento, al elemento que esté a la distancia designada por el otro operando. El resultado está enmascarado entre la nomenclatura designada en la Ilustración 4 para denotar a los sucesores, pero la solución es la correcta:

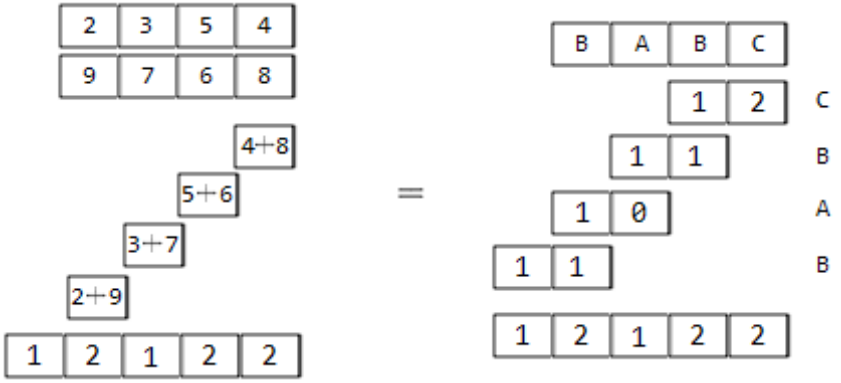

Ilustración 6. Correspondencia entre símbolos elementos sucesores y algoritmo.
Fuente: elaboración propia.

Tanto la Ilustración 5 como la Ilustración 6 muestran el mismo cálculo, pero en el primer caso el resultado requiere de un último paso final para deshacer la representación mostrada en la Ilustración 4 que sirve, en el punto en que està la redacción, para para explicar de una forma conceptualmente más exacta la transformación algebraica que se realiza.

Una vez destacado con este primer ejemplo el papel que juega el trabajar con el concepto de sucesor de un elemento de nuestra estructura algebraica, se pasa al segundo ejemplo, pero esta vez trabajando directamente con el concepto de operación de sucesor y el papel que juega en la representación por peso posición que se utiliza actualmente en los operandos.

Ahora se va a ejemplificar gráficamente el caso de la adición entre: 7008945 y 3567055:

Ed. 26. Vol.7 Nº 3. Septiembre-Diciembre 2018
DOI: http://dx.doi.org/10.17993/3ctic.2018.61.10-21/

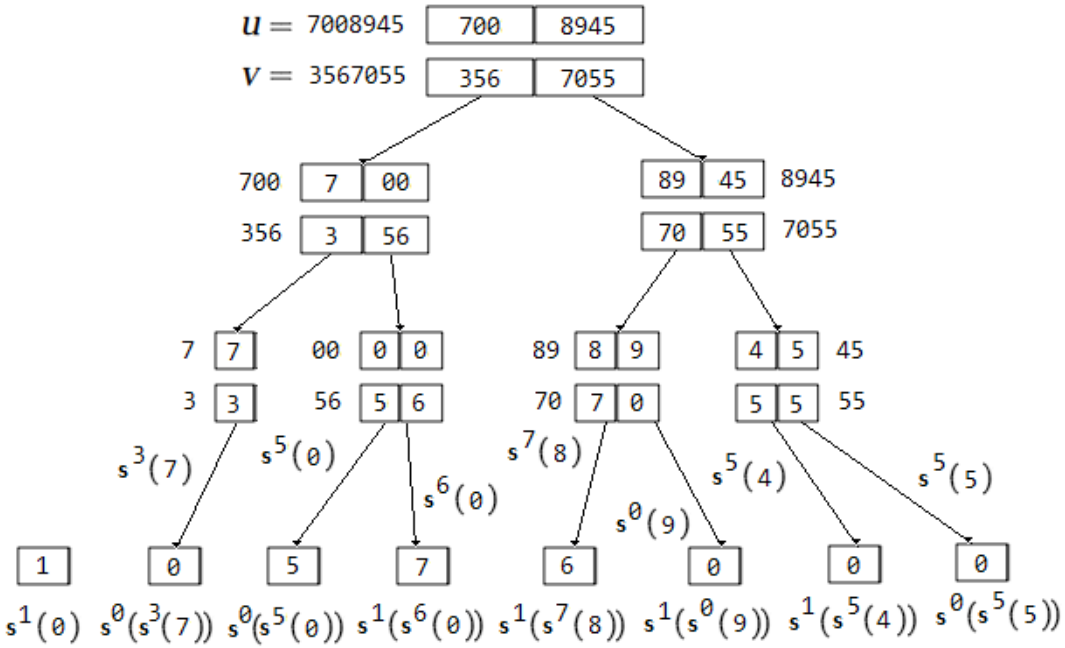

Ilustración 7. Técnica de particionado aditivo mediante Karatsuba, e.g. 7008945 + 3567055.
Fuente: elaboración propia.

En el nuevo caso, en la Ilustración 7 se nota mejor ese efecto de peso simbólico que es arrastrado en la representación de la información fruto de proyectar algunos elementos contra su sucesor con respecto a la relación ordinal definida en la estructura de grupo en la que se está trabajando (Ayuso, 2013-2018), para el presente desarrollo: un grupo abeliano.

Hay que entender para una comprensión correcta de lo expuesto que el cómputo que se buscar calcular en el fondo es:

$$s^{3567055}(7008945)$$

Fórmula 3. Cálculo en nomenclatura de sucesores para la **Ilustración 5.** 7008945 + 3567055.

Sin embargo, existe un estrecho vínculo entre estos algoritmos y la manera en la que es representada la información, normalmente con sistemas numéricos posicionales: mayormente el sistema arábigo. De ahí que esa proyección a elementos sucesores se acabe computando igualmente de manera posicional, por eficiencia, provocando ese comportamiento tradicionalmente comprendido como

acarreo, y que para este artículo a diferencia se concibe conceptualmente como una proyección ordinal de mayor magnitud (Booth, 1951) debido al peso significativo del dígito en el sistema numérico posicional.

Existe un estrecho vínculo entre estos algoritmos y la manera en la que es representada la información, normalmente con sistemas numéricos posicionales: mayormente el sistema arábigo.

Finalmente, también destacar cómo en la solución presentada se ha utilizado un tamaño de caso base: n igual a 1. En los algoritmos de este tipo, precisar el tamaño del caso base puede ser crucial de cara a explotar la eficiencia de la solución. Para los objetivos de este documento ese detalle quedaba fuera del ámbito del estudio que se ha realizado.

4. CONCLUSIONES

Aplicar el algoritmo de Karatsuba a otros contextos algebraicos permite extrapolar dicha herramienta algorítmica a un mayor número de escenarios, con la repercusión directa que esto supone: una alternativa más para abordar los cálculos, además de la capacidad de subdividir nuestra operación en problemas más pequeños. Con lo cual, se abre una puerta a paradigmas más susceptibles de un proceso de paralelización y a una mayor simplicidad en los cómputos a realizar.

En conclusión, el concepto propuesto por Karatsuba es extensible a distintas estructuras algebraicas, ofreciendo siempre la posibilidad de realizar un particionado de la operación, encajando de forma muy natural en un enfoque recursivo, y de esta manera reducir el problema a un conjunto de subproblemas de tamaño más reducido con una unión final del resultado de todos ellos: para la obtención del resultado al problema global.

El estudio, por otra parte, no sólo evidencia como ya se ha comentado esa adaptabilidad del concepto a otros contextos, con ese desarraigo del terreno en que fue concebido, sino que además proporciona una solución diferente para abordar el cómputo de la suma, e incluso la resta entre enteros de gran longitud poniendo a disposición las ya consabidas ventajas que ofrece esta idea en lo que a optimización y rendimiento se refiere, y dejando en entre dicho esos órdenes de complejidad tácitos para los algoritmos tradicionales.

Como posible futura línea de investigación, se pueden tratar de aplicar los conceptos descritos a entornos donde la operación aditiva se defina sobre un grupo abeliano diferente al tratado en el presente documento.

5. REFERENCIAS BIBLIOGRÁFICAS

Ayuso, J. (2015a). Booth algorithm operations addition and subtraction. *3C TIC*, 4(2), pp. 113-119.

Ayuso, J. (2015b). Booth algorithm modular arithmetic operations of addition and subtraction. *3C TIC*, 4(3), pp. 222-229.

Ayuso, J. (2015c). Booth algorithm modular arithmetic operations of multiplication. *3C TIC*, 4(4), pp. 255-221.

Ayuso, J. (2016a). Booth algorithm operations modular inverse. *3C TIC,* 5(2), pp. 28-41.

Ayuso, J. (2016b). Booth algorithm in signed-digit representation. *3C TIC,* 5(3), pp. 33-43.

Ayuso, J. (2017a). Booth algorithm in modular exponentiation operations. *3C TIC*, 6(2), pp. 1-12.

Ayuso, J. (2017b). Booth algorithm hardware operations addition and subtraction. *3C TIC*, 6(3), pp. 1-9.

Ayuso, J. (2017c). Booth algorithm in arity with multiple operands. *3C TIC,* 6(4), pp. 19-26.

Ayuso, J. (2018). Karatsuba algorithm operations exponentiation. *3C TIC,* 7(1), pp. 13-20.

Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, pp. 503–516.

Booth, A. D. (1945). A method of calculating reciprocal spacings for X-ray reflections from a monoclinic crystal. *J. Sci. Instr*, 22, pp. 74.

Booth, A. D. (1951). A signed binary multiplication technique. *Q. J. Mech. and Appl. Math.*, 4(2), pp. 236-240.

Booth, A. D. y Britten, K. H. V. (1947). *General Considerations in the Design of an Electronic Computer.*

Euclid of Alexandria. (1557). *Elements*. T.L. Heath's.

Karatsuba, A. y Ofman, Y. (1962). Multiplication of multidigit numbers on automata. *Doklady Akademii Nauk SSSR*, 145, pp. 293–294.

Newton, I. (1669). *De analysi per aequationes numero terminorum infinitas.*

Pascal, B. (1654). *Traité du Triangle Arithmétique.*