UNA INTERPRETACIÓN DE LA CONVOLUCIÓN ÚTIL PARA MULTIPLICAR ENTEROS DE LONGITUD ARBITRARIA CON LA FFT

Enric M. Calvo

Estudiante de la ETSETB y Miembro de la Rama de estudiantes del IEEE de Barcelona
enric25@casal.upc.es

INTRODUCCIÓN

En este artículo se va a intentar presentar una interpretación "evidente" de la convolución (es decir, evidente una vez ha sido explicada).

Después veremos qué utilidad podría tener, si es que tiene alguna, en la multiplicación de números enteros de longitud tan larga como se quiera, gracias a que la convolución de dos secuencias se convierte en un cálculo muy eficiente gracias a la *FFT - Fast Fourier Transform.*

Al mismo tiempo, se intentará presentar las circunstancias que hicieron posible encontrarla, dado que el autor considera interesante conocer de qué modo se producen estos hallazgos.

El único requisito para seguir con éxito este artículo es tener nociones básicas sobre teoría de sistemas lineales (saber qué es una convolución discreta, la *FFT*, ...).

ANTECEDENTES

Hace algún tiempo, mientras ojeaba un libro sobre curiosidades matemáticas vi una pequeña relación que me hizo gracia, de la que luego me acordaría durante algún tiempo. La curiosidad consistía sólo en ver que:

11*11=121
111*111=12321
1111*1111=1234321

Por otra parte, al cabo de algunos meses y después de llevar tratando con convoluciones discretas durante algún tiempo, aprecié, de un modo distinto, la convolución entre dos pulsos rectangulares iguales, que da como resultado un pulso triangular (resultado que todo el mundo conoce después de haber hecho una asignatura sobre señales y sistemas digitales o analógicos).

Como se puede observar, parece que tiene que haber alguna relación entre los dos hechos, así que sólo quedaba investigar de qué manera se relacionan la multiplicación de enteros con la convolución.

CONVOLUCIONAR ES MULTIPLICAR COMO SE APRENDE EN E.G.B.

Sólo hace falta pararse unos minutos a ver de qué forma se realiza la convolución cuando se calcula a mano para ver que, cuando realizamos a mano una multiplicación, estamos aplicando exactamente el mismo algoritmo (con la única diferencia que lo hacemos en módulo 10). Por ejemplo, si queremos multiplicar 34·67 sólo tenemos que crear dos vectores en orden inverso al visualizado, es decir:

$$x(n) = 4\delta(n) + 3\delta(n-1) = [\underline{4},3]$$
$$h(n) = 7\delta(n) + 6(n-1) = [\underline{7},3]$$

La convolución de ambos vectores da como resultado $y(n)$ = [$\underline{28}$, *45, 18*]. Alguien podría preguntarse en qué se parece este resultado con el esperado (34·67=2278). Pues bien, si tenemos en cuenta que en las secuencias que habíamos creado lo habíamos hecho en módulo 10, veamos qué ocurre si aplicamos ahora módulo 10 llevándonos el acarreo al siguiente dígito de mayor peso, es decir, el de la derecha. Entonces, la secuencia de salida después de llevarnos el acarreo de la primera posición (28 mod 10 = 8, luego nos llevamos 2 al siguiente) es $y(n)$ = [$\underline{8}$, *47, 18*]. Haciendo lo mismo para el segundo dígito: $y(n)$ = [$\underline{8}$, *7, 22*]. Y, por último, $y(n)$ = [$\underline{8}$, *7, 2, 2*], que es el número que esperábamos (si, del mismo modo que cuando hemos creado los vectores, leemos éste de derecha a izquierda).

Puede parecer que todo esto es una "idea feliz", o una simple casualidad, pero ahora veremos que no es así, que tiene una razón de ser.

AHORA CON TRANSFORMADA Z

Seguramente la manera más rigurosa, bonita y fácil de ver que el algoritmo de multiplicar que se aprende con 7 años es el mismo que el que se aplica en la convolución, es que observemos los números como polinomios en s para luego hacer lo mismo con las secuencias transformadas ya en polinomios de variable z^{-1} (después de haber aplicado la Transformada Z). La gracia de multiplicar los polinomios después de transformarlos es que sabemos que esto equivale a convolucionar las secuencias en tiempo, y acabamos de ver que haciéndolo estaremos realizando la multiplicación que deseábamos. Será, por tanto, la manera más "natural" de tratar con los números que queremos multiplicar, ya que multiplicar polinomios sabemos hacerlo.

El número 34 puede verse como un polinomio $x(s) = 3s + 4$ si se evalúa en $s = 10$. El 67 sería: $h(s) = 6s + 7$. Multiplicados ambos polinomios tenemos $y(s) = x(s) \cdot h(s) = 18s^2 + 45s + 28$, que evaluado en s = 10 da el resultado esperado: 34·67 = 2278.

Sólo hace falta substituir $s = z^{-1}$ en los anteriores polinomios y veremos que conseguimos las transformadas Z de las secuencias de ejemplo que hemos utilizado en el apartado anterior:

$$X(z) = 4 + 3z^{-1}$$
$$H(z) = 7 + 6z^{-1}$$

En este caso encontramos que los dos polinomios multiplicados, al dar el mismo resultado que convolucionando las secuencias, resultan:

$$Y(z) = X(z)H(z) = 28 + 45z^{-1} + 18z^{-2}$$

Por supuesto, igual que en el caso de polinomios en s substituyendo $z^{-1} = 10$ encontraremos el resultado esperado. Pero, del mismo modo que cuando convolucionábamos, si queremos como resultado una secuencia que sea comprensible a primera vista (es decir, con un dígito en cada posición y ordenados de menor a mayor peso según crece n), tenemos que hacer módulo diez en cada coeficiente y llevarnos el acarreo al siguiente dígito de mayor peso. Encontraríamos entonces que el polinomio "maquillado" es:

$$Y'(z) = 8 + 7z^{-1} + 2z^{-2} + 2z^{-3}$$

que antitransformado da el esperado:

$$y'(n) = [\underline{8}, 7, 2, 2]$$

APLICACIONES

Alguien puede preguntar de qué nos va a servir darnos cuenta de todo esto y qué utilidad tiene para multiplicar números. Pues lo bueno de todo este asunto es lo siguiente: por un lado acabamos de demostrar que multiplicar dos números enteros es equivalente a convolucionar dos secuencias adecuadamente creadas. Por otro lado sabemos que convolucionar se hace de la forma más rápida con la FFT (cuando N es un poco grande). Mezclando estas dos ideas surge la de tratar de multiplicar números de longitud arbitraria con este método.

En efecto, dado que $x(n)*h(n)=y(n)$ equivale a $y(n) = IFFT\{X(k)H(k)\}$ podemos multiplicar dos cifras con este método. Pero, no sólo eso, si pensamos en el problema inverso (lo que sería un problema de *deconvolución*), podemos, además dividir usando este método ($H(k)=Y(k)/X(k)$), aunque ésta última operación conlleva más problemas.

Ejemplo de un multiplicación 568974568712345 · 458796325874125=261043441641038763007803573125 con *FFT*:

x(n) = [5, 4, 3, 2, 1, 7, 8, 6, 5, 4, 7, 9, 8, 6, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

h(n) = [5, 2, 1, 4, 7, 8, 5, 2, 3, 6, 9, 7, 8, 5, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

$$y(n) = x(n) * h(n)$$

$$y(n) = IFFT\{X(k)Y(k)\}$$

y(n) = [25, 30, 28, 40, 63, 119, 141, 125, 134, 183, 271, 321, 328, 320, 352, 380, 402, 406, 376, 342, 308, 287, 282, 263, 224, 159, 102, 49, 20, 0, 0, 0]

Haciendo una pasada de acarreos queda:

y(n) = [5, 2, 1, 3, 7, 5, 3, 0, 8, 7, 0, 0, 3, 6, 7, 8, 3, 0, 1 ,4, 6, 1, 4, 4, 3, 4, 0, 1, 6, 2]

Notar que se les ha hecho un *zero-padding* a ambos vectores hasta N=32, para así poder hacer cómodamente la IFFT.

PUNTOS FUERTES

Algunos rasgos atractivos de este método son, por ejemplo:

· Realizar FFT's es muy rápido para secuencias con $N = 2^n$, luego se puede conseguir la multiplicación de 2 secuencias larguísimas en muy poco tiempo.

· Este método es igualmente aplicable para otras bases. Así, es trivial implementarlo en base 16, con lo que el acarreo, por ejemplo, no supondría apenas coste computacional.

· Viendo que es válido para números enteros, también lo es para números reales (de precisión finita). Para ello sólo hay que considerar que la secuencia de salida tiene una coma situada en la posición resultante de sumar el número de decimales de las dos secuencias.

La aplicación más notable para la que pensé que podía llegar a ser útil es en criptografía, dado que los números con los que se trabaja son larguísimos (unos 155 dígitos en decimal equivalen a una clave de 512 bits, aprox.), aunque, dado que la eficiencia de este método crece conforme N aumenta, es posible que sea para un N bastante grande cuando multiplicar usando métodos basados en FFT sea más eficiente que con métodos tradicionales. Tal vez sería útil para la factorización de números enormes en dos primos, aunque como veremos dividir no resulta tan óptimo como multiplicar.

INCONVENIENTES Y OTRAS CONSIDERACIONES A TENER EN CUENTA

La verdad es que para comprobar la utilidad de este método haría falta saber el coste computacional de otros métodos de multiplicación que se utilizan para manejar números de longitud arbitraria, ya sea implementados vía software o vía hardware, costes que en el momento de redactar el artículo el autor desconocía. La *FFT* tiene un coste computacional de $(N/2)log(N)$ multiplicaciones, con lo que invalidaría la utilidad del método si existe actualmente una implementación con menor coste computacional.

También se presentan varios problemas básicos relacionados con la longitud de las secuencias a multiplicar y la FFT.

· Sabemos que, de hecho, la multiplicación de las FFT's de dos secuencias da como resultado la convolución *circular*, y no la convolución normal. Para solucionar este inconveniente, sólo hace falta rellenar las secuencias con 0's (*zero-padding*) hasta tener 2 secuencias de longitud $N = 2^n$ cada una, donde N sea más grande que la suma de las longitudes en dígitos de las dos cifras a multiplicar.

· El mayor inconveniente de este método es que, una vez transformadas las secuencias tenemos que multiplicar (o dividir) punto a punto cada secuencia. Éste es uno de los mayores "handicaps" de este método, ya que requiere la multiplicación de 2 números complejos (donde se necesitan 4 multiplicaciones reales), o la división de 2 números complejos (donde pueden llegar a necesitarse 6 multiplicaciones y 2 divisiones reales, si es que no hay algún método óptimo). De todos modos en el apartado de optimizaciones veremos qué podemos hacer para no tener que multiplicar punto a punto.

· Otro inconveniente que habría que comprobar si es importante es que, debido a que *N* puede ser muy grande, no haya un arrastre de errores en la *FFT* y en la *IFFT* que haga que los resultados a partir de una cierta *N* sean equivocados. Hay que tener en cuenta que cada dígito está representado por un valor de 16 ó 32 bits, por lo que hay que asegurarse que no desborda. El peor caso sería multiplicar dos secuencias de N valores máximos cada una, y el mayor resultado se daría cuando, en la convolución, no hubiese desfase entre ambas (es decir, justo el instante de la convolución en que las dos secuencias "coinciden"). El valor del dígito en ese punto sería de $N{\cdot}(base-1)^2$. Podemos calcular entonces fácilmente la *N* máxima calculable para los casos en que utilicemos enteros de 16, y de 32 bits y usemos base 10 ó 16 (ver tabla adjunta).

	base 10	base 16
int16	809	256
int32	53*10^6	16777216

***Tabla 1**: nº máximo de dígitos*

· Dado que después de aplicar la *FFT* tenemos una secuencia compleja compuesta por un par de números reales de precisión finita, es posible que al antitransformar los números no sean enteros, con lo que habría que tratar de acotar el error de algún modo.

OPTIMIZACIONES

A la vista de la operación que hay que hacer para calcular la DFT (aunque a la práctica se realizase con la *FFT*)

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi kn}{N}}$$

podemos proponer alguna "optimización". Por ejemplo para calcular la *DFT* de $x(n)$. Se puede observar que, como hemos dispuesto un dígito de la cifra en cada posición del vector $x(n)$, en este sumatorio $x(n)$ no tendrá otro valor que uno comprendido en el rango *0-9*. Esto da naturalmente pie a que el cálculo de la *DFT* pueda reducirse a la suma de N referencias a una tabla de *10*N* complejos. Esta tabla puede incluso caber en la *caché* de un ordenador, con lo que nos aseguramos una velocidad muy alta de acceso a esta tabla. Naturalmente esta optimización puede ser mayor si descartamos el *0* y nos damos cuenta que con los valores de un *cos(x)* evaluado entre $[0, \frac{\pi}{2}]$ tenemos suficiente para representar todos los valores de las funciones trigonométricas que componen la exponencial compleja. Con estas últimas optimizaciones podríamos llegar a necesitar sólo $\frac{9N}{4}$ referencias. El inconveniente de optimizar la *DFT* (no la *FFT*) es que tenemos que hacer $2N^2$ sumas, cosa que no asegura una mejora rotunda del algoritmo en comparación con hacer algunas multiplicaciones.

Otra optimización podría basarse en una de las ventajas antes comentadas. Dado que la base a utilizar en la secuencia es arbitraria, podríamos usar base 16, una más natural para los ordenadores. De este modo el acarreo (dividir módulo 16) pasa a ser una secuencia muy simple de lógica binaria, cosa que parece muy conveniente. Pero aún se puede llevar más allá, de manera que si los dígitos con los que trabajamos son todavía mayores (siempre en base 2^n) podemos ahorrarnos varias multiplicaciones para hacer el mismo cálculo (es como si hubiésemos "concentrado" en menos coeficientes la "señal").

CONCLUSIONES

Bien, decía al principio del artículo que convolucionar era como multiplicar enteros con el algoritmo que todo niño conoce, aunque, para ser exactos habría que decir que es igual que multiplicar polinomios. Esto es evidente, en cuanto a que convolucionar dos secuencias es lo mismo que multiplicar sus transformadas Z, que no dejan de ser polinomios. Después hemos aplicado el hecho de que convolucionar se hace más rápido en frecuencia, con lo que nos ahorramos cálculos. Como puede verse, al fin y al cabo era algo evidente lo que se trataba de hacer notar en este artículo, aunque tal vez no todo el mundo se había dado cuenta de ello.

Futuras líneas de investigación podrían ir encaminadas a estudiar el mismo problema pero con otras transformadas, para ver si es posible ahorrarnos los cálculos en complejos que implica el trabajar con la *FFT*.

También se ha expuesto el hecho de que es muy posible que otros métodos para el cálculo de multiplicaciones y divisiones de números de longitud arbitraria sea más eficiente, o que con este método se produzcan errores a partir de un cierto valor *N*. Si es así y el método no es válido, esta interpretación de la convolución no pasará más allá de ser una curiosidad bella de admirar.