

Algorítmica

Capítulo 6. Otras Metodologías Algorítmicas.
Tema 17. Algoritmos de Transformación Algebraica
-La Transformada de Fourier

Parecía una locura

- Diseñar un algoritmo que es una unidad de magnitud mas rápido que otro, es un importante logro.
- Cuando esa mejora esta asociada a algún proceso que tiene muchas aplicaciones, lo conseguido tiene gran repercusión en el terreno científico y tecnológico.
- Este es el caso de la transformada de Fourier.
- Ninguna mejora en ningún algoritmo ha tenido un impacto mayor que el provocado por este método.
- ¿Por qué?. ¿Cuál es su origen?. ¿Tiene alguna repercusión en el contexto de la Informática?



¿Como se representan las señales?

- Una serie de Taylor series representa cualquier función empleando polinomios

$$\begin{aligned}f(x) &= f(\alpha) + f'(\alpha)(x - \alpha) + \frac{f''(\alpha)}{2!} \\&\quad (x - \alpha)^2 + \frac{f^{(3)}(\alpha)}{3!} (x - \alpha)^3 + \dots + \frac{f^{(n)}(\alpha)}{n!} (x - \alpha)^n + \dots\end{aligned}$$

- Pero los polinomios dan problemas porque son inestables y porque fisicamente no son fácilmente comprensibles.
- Nos resulta mas intuitivo y sencillo hablar sobre “señales” en terminos de sus “frecuencias” (como de rápido, cada cuanto cambian las señales, etc.)

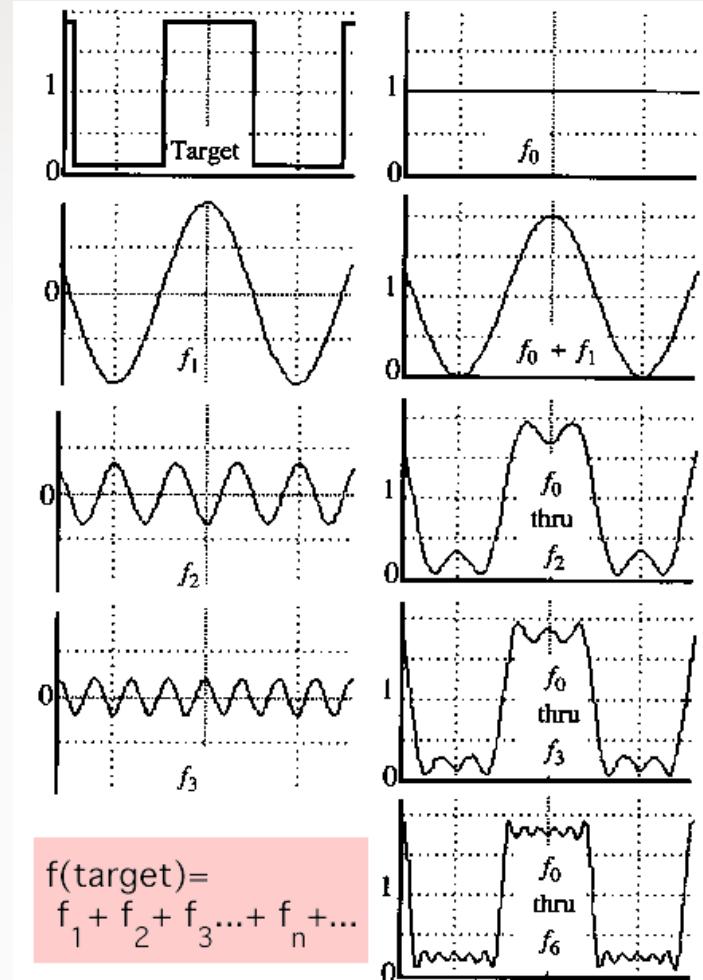
Jean Baptiste Joseph Fourier (1768-1830)

- En 1807 Fourier tuvo una idea:
- Cualquier función periódica podía reescribirse como la suma ponderada de senos y cosenos de diferentes frecuencias
- Difícil de creer ¿verdad?
 - No se lo creyeron Lagrange, Laplace, Poisson y muchos otros “popes” de la época.
 - Su trabajo no se tradujo al inglés hasta 1878 (¡71 años después!)
- Pero es verdad
 - Se hace con la **Serie de Fourier**
 - Posiblemente la mas poderosa herramienta de las Ingenierías



Una suma de sinusoides

- A partir de la herramienta básica:
 $A \sin(\omega x + \phi)$
- Como en el mundo de los bloques:
- se suman tantos de ellos como sean necesarios (se construye la serie) para representar la señal $f(x)$ que queremos reproducir



La transformada de Fourier

- Buscamos interpretar (conocer) la frecuencia ω de nuestra señal. Para ello reparametrizamos la señal, en lugar de por medio de x , por medio de ω



- Para cualquier ω entre 0 e infinito, $F(\omega)$ nos da la amplitud A y la fase ϕ de $A \sin(\omega x + \phi)$
- Esto es así gracias a que,

$$F(\omega) = R(\omega) + iI(\omega)$$

$$A = \pm \sqrt{R(\omega)^2 + I(\omega)^2} \quad \phi = \tan^{-1} \frac{I(\omega)}{R(\omega)}$$



La Transformada Rápida de Fourier

- La transformada de Fourier de una función continua $a(t)$ esta dada por

$$A(f) = \int_{-\infty}^{\infty} a(t)e^{2\pi i f t} dt$$

- mientras que la transformada inversa esta dada por

$$a(t) = 1/(2\pi) \int_{-\infty}^{\infty} A(f)e^{-2\pi i f t} df.$$

- La i en las dos anteriores ecuaciones representa la raiz cuadrada de -1 y la constante e es la base de los logaritmos naturales.
- A menudo la variable t se entiende que es el tiempo, mientras que f se toma como la frecuencia, de modo que, como hemos explicado antes,
- La transformada de Fourier expresa una función del tiempo en una de la frecuencia.

La Transformada Rápida de Fourier

- Asociada a esta transformada de Fourier continua, hay una versión discreta, que se llama **Transformada de Fourier Discreta**, y que trabaja sobre puntos muestrales de $a(t)$, a_0 , a_1 , ... a_{N-1} .
- La transformada discreta de Fourier se define por

$$A_j = \sum_{0 \leq k \leq N-1} a_k e^{2\pi i j k / N}, \quad 0 \leq j \leq N - 1$$

- siendo la inversa

$$a_k = (1/N) \sum_{0 \leq j \leq N-1} A_j e^{-2\pi i j k / N}, \quad 0 \leq k \leq N - 1$$

- En el caso discreto se da un conjunto de N puntos muestrales y se obtiene un nuevo conjunto de N puntos.

La Transformada Rápida de Fourier

- Llegado a este punto nos interesa destacar la estrecha conexión que existe entre la Transformada Discreta de Fourier y la evaluación de polinomios.
- En efecto, supongamos el polinomio

$$a(x) = a_{N-1}x^{N-1} + a_{N-2}x^{N-2} + \dots + a_1x + a_0$$

- entonces el elemento Fourier A_j es el valor de $a(x)$ en $x = w^j$, donde $w = e^{2\pi i / N}$. De manera similar, para la transformada inversa de Fourier, si imaginamos el polinomio con los coeficientes de Fourier

$$A(x) = A_{N-1}x^{N-1} + A_{N-2}x^{N-2} + \dots + A_1x + A_0$$

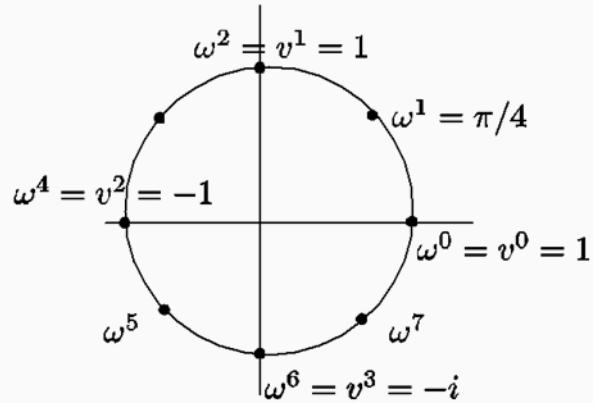
- entonces cada a_k es el valor de $A(x)$ en $x = (w^{-1})^k$ donde $w = e^{2\pi i / N}$.
- Por tanto la Transformada de Fourier Discreta se corresponde exactamente con la evaluación de un polinomio en N los puntos w^0, w^1, \dots, w^{N-1} .

La Transformada Rápida de Fourier

- Recordemos que un polinomio de grado N podemos evaluarlo en N puntos usando $O(N^2)$ operaciones.
- Aplicamos el algoritmo de Horner una vez en cada punto.
- La Transformada Rápida de Fourier (TRF) es un algoritmo para calcular esos N valores usando solo $O(N \log N)$ operaciones.
- Este algoritmo fue popularizado por Cooley y Tukey en 1965, y luego explotado por el mismo Cooley.
- Conviene fijarse en que la Transformada de Fourier puede ser mas rápida que el algoritmo de Horner porque los puntos de evaluación no son arbitrarios, sino mas bien muy especiales. De hecho son las N potencias w^j para $0 \leq j \leq N-1$, donde $w = e^{2\pi i/N}$.
- El punto w es una raiz primitiva N -ésima de la unidad en el plano complejo

Formalidades

- **Definición.** Un elemento w en un anillo commutativo es llamado una raíz primitiva N -ésima de la unidad si
 - (i) $w \neq 1$
 - (ii) $w^N = 1$
 - (iii) $\sum_{p=0..N-1} w^{jp} = 0, 1 \leq j \leq N-1$
- De manera más intuitiva, una raíz N -ésima de la unidad es un número complejo z tal que $z^n = 1$. Exactamente hay N raíces primitivas de la unidad, $w^k, k = 0, 1, \dots, N-1$



Formalidades

- Las dos siguientes propiedades de las raíces N-ésimas sirven para entender mejor el algoritmo de la TRF.
 - 1) Sea $N = 2n$ y supongamos que w es una raíz primitiva N-ésima de la unidad. Entonces
$$- w^j = w^{j+n}$$
 - 2) Sea $N = 2n$ y w una raíz primitiva N-ésima de la unidad. Entonces w^2 es una raíz primitiva N-ésima de la unidad.
- A partir de estas propiedades se puede concluir que si w^j , $0 < j \leq N-1$ son las raíces primitivas N-ésimas de la unidad, $N = 2n$, entonces w^{2j} , $0 < j \leq n-1$ son n raíces primitivas de la unidad.
- A partir de estas dos propiedades podemos obtener un algoritmo Divide y Vencerás para la Transformada de Fourier.
- La complejidad del algoritmo será $O(N \log N)$, mas rápido por tanto que el que proporciona el algoritmo de evaluación polinómica convencional $O(N^2)$.

Algoritmo para el cálculo de la TRF

- Consideremos de nuevo que los coeficientes que queremos transformar son a_{N-1}, \dots, a_0 y sea

$$a(x) = a_{N-1}x^{N-1} + \dots + a_1x + a_0$$

- Dividimos $a(x)$ en dos partes, una conteniendo los exponentes numerados impar, y la otra con los pares.

$$\begin{aligned} a(x) = & a_{N-1}x^{N-1} + a_{N-3}x^{N-3} + \dots + a_1x + \\ & + a_{N-2}x^{N-2} + \dots + a_2x^2 + a_0 \end{aligned}$$

- Tomando $y = x^2$ podemos reescribir $a(x)$ como la suma de dos polinomios

$$\begin{aligned} a(x) = & (a_{N-1}y^{n-1} + a_{N-3}y^{n-2} + \dots + a_1)x + (a_{N-2}y^{n-1} + a_{N-4}y^{n-2} + \dots + a_0) = \\ = & c(y) \cdot x + b(y) \end{aligned}$$

Algoritmo para el cálculo de la TRF

- $a(x) = (a_{N-1}y^{n-1} + a_{N-3}y^{n-2} + \dots + a_1)x + (a_{N-2}y^{n-1} + a_{N-4}y^{n-2} + \dots + a_0) =$
 $= c(y) \cdot x + b(y)$
- Recordemos que los valores de la TRF son $a(w^j)$, $0 \leq j \leq N-1$ y que los valores de $a(x)$ en los puntos w^j , $0 \leq j \leq n-1$ son ahora expresables como,

$$a(w^j) = c(w^{2j}) w^j + b(w^{2j})$$

$$a(w^{j+n}) = -c(w^{2j}) w^j + b(w^{2j})$$

- Con estas dos formulas un problema de tamaño N se resuelve transformándolo en 2 problemas idénticos de tamaño $n = N/2$.
- Estos subproblemas son la evaluación de $b(y)$ y $c(y)$, cada uno de grado $n-1$, en los puntos $(w^2)^j$, $0 \leq j \leq n-1$, que son raíces primitivas n -ésimas.

Algoritmo para el cálculo de la TRF

- Esto es un ejemplo de Divide y Vencerás, y por tanto podemos volver a aplicar esa metodología mientras sigamos teniendo un número par de puntos.
- Esto nos lleva a elegir siempre N como una potencia de 2, $N = 2^m$, ya que así podemos continuar desarrollando este procedimiento de división hasta que alcancemos un problema trivial, es decir, hasta que lleguemos a tener que evaluar un polinomio constante.
- Así, el algoritmo Divide y Vencerás para el cálculo de la TRF tiene una eficiencia $O(N \log N)$ obtenida a partir de la recurrencia,

$$T(N) = 2T(N/2) + cN$$

Algoritmo para el cálculo de la TRF

- Un algoritmo recursivo DV para el cálculo de la TRF (Horowitz y Sahni, 1978) es este:

```
procedure FFT( $N$ ,  $a(x)$ ,  $w$ ,  $A$ )
    // $N = 2^m$ ,  $a(x) = a_{N-1}x^{N-1} + \dots + a_0$ ,  $w$  is a //
    //primitive  $N$ -th root of unity  $A(0:N - 1)$  is set to //
    //the values  $a(w^j)$ ,  $0 \leq j \leq N - 1$ . //
    integer  $N$  real  $A(0:N - 1)$ ,  $B(0:(N/2) - 1)$ ,  $C(0:(N/2) - 1)$ ,
     $WP(-1:(N/2) - 1)$ 
    if  $N = 1$  then  $A(0) \leftarrow a_0$ 
    else  $n \leftarrow N/2$ 
         $b(x) \leftarrow a_{N-2}x^{n-1} + \dots + a_2x + a_0$  //divide the coefficients //
         $c(x) \leftarrow a_{N-1}x^{n-1} + \dots + a_3x + a_1$  //into 2 sets //
        call FFT( $n$ ,  $b(x)$ ,  $w^2$ ,  $B$ ) //apply this algorithm again //
        call FFT( $n$ ,  $c(x)$ ,  $w^2$ ,  $C$ ) //and again //
         $WP(-1) \leftarrow 1/w$ 
        for  $j = 0$  to  $n - 1$  do
             $WP(j) \leftarrow w * WP(j - 1)$ 
             $A(j) \leftarrow B(j) + WP(j) * C(j)$ 
             $A(j + n) \leftarrow B(j) - WP(j) * C(j)$ 
        repeat
    endif
end FFT
```

Una última nota

- Consideraremos el problema de la multiplicación de polinomios.
- La técnica de la transformada llama a evaluar $A(x)$ y $B(x)$ en $2N + 1$ puntos, calculando los $2N + 1$ productos $A(x_i) \cdot B(x_i)$ y obteniendo el producto $A(x)B(x)$.
- Sabremos que la evaluación de N puntos requiere $O(N^2)$ operaciones, de modo que parece que no ganamos nada con esta técnica de la transformada.
- Pero lo que acabamos de ver es que si los puntos los elegimos de manera que sean las $N = 2^m$ potencias distintas de una raíz primitiva N -ésima de la unidad, entonces la evaluación, y la interpolación, pueden hacerse con $O(N \log N)$ operaciones.
- Por tanto usando el algoritmo de la TRF podemos multiplicar dos polinomios de grados N con $O(N \log N)$ operaciones.