

CSE373: Estructuras de datos y algoritmos

Divide y conquistarás:

La transformada rápida de Fourier

steve tanimoto
Otoño 2016

Aspectos destacados de la transformada rápida de Fourier


- Podría decirse que es el algoritmo numérico práctico más notable para el análisis de datos.
- Utiliza el paradigma divide y vencerás.
- También utiliza la técnica de programación dinámica.
- Por lo general, utiliza la recursividad.
- Aprovecha la algo misteriosa ecuación de Euler que sorprendentemente relaciona algunas de las constantes matemáticas más importantes de una manera concisa:

$$mi + 1 = 0$$
- Es rápido y tiene tiempo de ejecución en $\Theta(n \log n)$.
- Hay muchas variaciones, a menudo para aprovechar la estructura numérica de n . Más comúnmente, n es una potencia de 2.
- También se puede calcular ópticamente usando lentes.

Otoño 2016 CSE 373: Estructuras de datos y algoritmos 2

Transformadas de Fourier

- Joseph-A Fourier observó que cualquier función continua $f(x)$ puede expresarse como una suma de funciones seno - $\sin(-x + \cdot)$, cada una de ellas adecuadamente amplificada y desfasada.
- Su objeto era caracterizar la tasa de transferencia de calor en los materiales.
- La transformada nombrada en su honor es una técnica matemática que se puede utilizar para el análisis, compresión y síntesis de datos en muchos campos.



Otoño 2016 CSE 373: Estructuras de datos y algoritmos 3

Definición

- Dejar $f = f_0, \dots, f_{n-1}$ sea un vector de n números complejos.
- La transformada discreta de Fourier de f es otro vector de n números complejos $F = F_0, \dots, F_{n-1}$ cada uno dado por:

$$F_k = \sum_{j=0}^{n-1} f_j e^{(-\frac{2\pi i j k}{n})}$$

- Aquí $i = \sqrt{-1}$ (la unidad imaginaria)

Otoño 2016 CSE 373: Estructuras de datos y algoritmos 4

Raíces enésimas de la unidad

- Los factores $e^{(-\frac{2\pi i j k}{n})}$ son raíces enésimas de la unidad:
- Son soluciones de la ecuación $x^n = 1$.
- Definir $\omega = e^{(-\frac{2\pi i}{n})}$
- Esta es una raíz n -ésima principal de la unidad, lo que significa que si $k \neq 1$ entonces k es múltiplo de n .
- Todos los demás factores son potencias de ω . Solo hay n potencias distintas que son relevantes cuando se procesa un vector de longitud n .

Otoño 2016 CSE 373: Estructuras de datos y algoritmos 5

Exponenciales complejas como ondas

- $mi = \text{porque} + \text{yo pecado}$ ("Fórmula de Euler", no identidad de Euler) La parte real de $e^{i\cdot}$ es una onda coseno. La parte imaginaria de $e^{i\cdot}$ es una onda sinusoidal.
- $real(e^{i\cdot}) = \text{porque}$
- $imag(e^{i\cdot}) = \text{pecado}$

Otoño 2016 CSE 373: Estructuras de datos y algoritmos 6

Exponenciales complejas como ondas

- $m_i =$ porque - + yo pecado -
 - $\text{real}(e^{i\cdot}) =$ porque -
 - $\text{imagen}(e^{i\cdot}) =$ pecado - -
- ("Fórmula de Euler", no identidad de Euler) La parte real de $e^{i\cdot}$ es una onda coseno. La parte imaginaria de $e^{i\cdot}$ es una onda sinusoidal.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

7

Exponenciales complejas como ondas

- $m_i =$ porque - + yo pecado -
 - $\text{real}(e^{i\cdot}) =$ porque -
 - $\text{imagen}(e^{i\cdot}) =$ pecado - -
- ("Fórmula de Euler", no identidad de Euler) La parte real de $e^{i\cdot}$ es una onda coseno. La parte imaginaria de $e^{i\cdot}$ es una onda sinusoidal.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

8

Exponenciales complejas como ondas

- $m_i =$ porque - + yo pecado -
 - $\text{real}(e^{i\cdot}) =$ porque -
 - $\text{imagen}(e^{i\cdot}) =$ pecado - -
- ("Fórmula de Euler", no identidad de Euler) La parte real de $e^{i\cdot}$ es una onda coseno. La parte imaginaria de $e^{i\cdot}$ es una onda sinusoidal.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

9

La DFT como transformación lineal

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^4 & \omega^6 \\ \omega^0 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

10

Cálculo de la transformada discreta de Fourier

$$F_k = \sum_{j=0}^{n-1} f_j e^{(-2\pi i j k / n)}$$

Método directo:

Suponga que las exponenciales complejas están precalculadas.

nóte: multiplicaciones complejas
 $n(n-1)$ sumas complejas

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

11

Divide y conquistas

- Divida el problema en subproblemas más pequeños, resuélvalos y combínelos en la solución general
- Resolver subproblemas recursivamente o con un método directo
- Combinar los resultados de los subproblemas
- Aplicar programación dinámica, si es posible

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

12

Una transformada rápida de Fourier recursiva

```
def FFT(f):
    n = largo(f)
    si n==1: devuelve [f(0)] F = # caso base
    n*=2 # Inicializa los resultados a 0.
    f_par = f[0::2] # Dividir - incluso subproblema.
    f_impar = f[1::2] # "- subproblema impar
    F_par = FFT(f_par) # llamada recursiva
    F_impar = FFT(f_impar) # "
    n2 = entero(n/2) # Prepárate para combinar resultados
    para i en el rango (n2):
        giro = exp(-2*pi*1j)*1/n término impar = # Estos podrían ser precalculados
        F_impar[i] * giro F[i] = F_par[i] + término # Los términos impares necesitan un ajuste
        impar F[i+n2] = F_par[i] - término impar # Calcular un nuevo término
    devuelve F # Calcular un nuevo término más
```

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

13

Un algoritmo $N \log N$

Al igual que en el ordenamiento por combinación, en cada llamada recursiva, dividimos el número de elementos por la mitad.

Por lo tanto, el número de niveles de llamadas recursivas es $\log_2 N$. Cuando combinamos los resultados de los subproblemas, gastamos tiempo lineal. El tiempo total está acotado por $c N \log N$.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

14

Desenrollando la FFT

(vistas más detalladas
de cómo funciona una FFT)

FFT recursiva

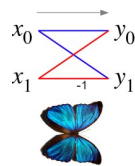
FFT(n , [a_0, a_1, \dots, a_{n-1}]):
 si $n=1$: devuelve uno
 Fincluso = FFT($n/2$, [u_0, a_2, \dots, a_{n-2}]) F
 extraño = FFT($n/2$, [u_1, a_3, \dots, a_{n-1}])
 para $k = 0$ a $n/2 - 1$:
 $\omega_k = \text{mlzmlk}/n$
 $y_k = \text{Fincluso } k + \omega_k \text{Fimpar } k$
 $y_{k+n/2} = \text{Fincluso } k - \omega_k \text{Fimpar } k$
 volver [y_0, y_1, \dots, y_{n-1}]

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

dieciséis

El paso de la mariposa



Un diagrama de flujo de datos que conecta las entradas X (izquierda) a las salidas que dependen de ellos (derecha) para un paso de "mariposa" de una FFT radix-2 Cooley-Tukey. Este diagrama se asemeja a una mariposa.

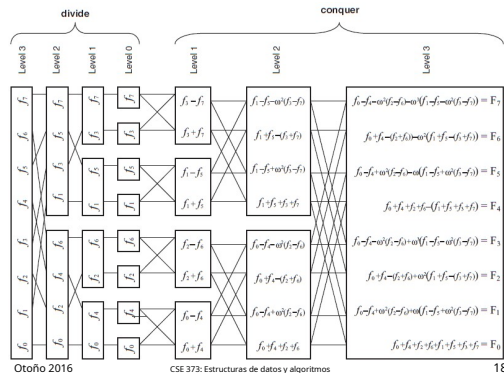
http://en.wikipedia.org/wiki/Butterfly_diagram

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

17

Recursión desenrollada



Otoño 2016

CSE 373: Estructuras de datos y algoritmos

18

Comentarios

- La FFT se puede implementar:
 - De forma iterativa, en lugar de recursivamente.
 - En el lugar, (después de poner la entrada en orden de bit invertido)
- Este diagrama muestra una FFT de "diezmación en el tiempo" de Radix-2, Cooley-Tukey.
- Usando una implementación de radix-4, la cantidad de multiplicaciones y sumas escalares se puede reducir entre un 10 y un 20 por ciento.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

19

FFT en la práctica

Hay muchas variedades de transformadas rápidas de Fourier. ellos típicamente dependen del hecho de que N es un número compuesto, como una potencia de 2.

No es necesario que la raíz sea 2, y se pueden usar raíces mixtas. Las formulaciones pueden ser recursivas o iterativas, en serie o en paralelo, etc.

También hay computadoras analógicas para transformadas de Fourier, como los basados en las propiedades de la lente óptica.

La transformada rápida de Fourier de Cooley-Tukey a menudo se considera el algoritmo numérico más importante jamás inventado. Este es el método al que normalmente se hace referencia con el término "FFT".

La FFT también se puede utilizar para convolución rápida, polinomio rápido multiplicación y multiplicación rápida de números enteros grandes.

Otoño 2016

CSE 373: Estructuras de datos y algoritmos

20