*Szybka Transformata Fouriera

Dariusz Borkowski

A

lgorytm wyznaczania DFT zwany szybką transformatą Fouriera został opublikowany w 1965 r. przez J. W. Colley'a oraz J. W. Tuckey'a. Sam algorytm oraz liczne jego modyfikacje pozwalają na wyznaczenie DFT dla N, które jest potęgą 2, i przy tym założeniu dają $N/\log_2(N)$-krotne przyspieszenie w porównaniu do algorytmu wynikającego bezpośrednio z definicji DFT. Aby zilustrować efektywność algorytmu FFT załóżmy, że $N = 1024$. Wówczas FFT daje wynik około 100 razy szybciej niż algorytm wynikający bezpośrednio z definicji.

Żeby zrozumieć mechanizm działania algorytmu FFT rozważmy przypadek dla $N = 8$. Korzystając z notacji $x(m) = f(x_m)$, $\omega_N = \exp(-2i\pi/N)$ dyskretną transformatę Fouriera można zapisać jako dwie sumy: jedną dla indeksów parzystych, a drugą dla nieparzystych.

$$\begin{aligned} c_k =& \frac{1}{N}\left(x(0) + x(2)\omega_8^{2k} + x(4)\omega_8^{4k} + x(6)\omega_8^{6k}\right) + \\ &+ \omega_8^k \frac{1}{N}\left(x(1) + x(3)\omega_8^{2k} + x(5)\omega_8^{4k} + x(7)\omega_8^{6k}\right) = \\ =& \frac{1}{N}\left(x(0) + x(2)\omega_4^{k} + x(4)\omega_4^{2k} + x(6)\omega_4^{3k}\right) + \\ &+ \omega_8^k \frac{1}{N}\left(x(1) + x(3)\omega_4^{k} + x(5)\omega_4^{2k} + x(7)\omega_4^{3k}\right). \end{aligned} \tag{1}$$

W pierwszym kroku DFT długości 8 jest zastąpiona dwiema transformatami długości 4. Powta-

rzając ten proces, w każdym kroku otrzymujemy dwa razy więcej DFT o połowę krótszych od poprzedników. I tak np. dla DFT długości 1024 po 10 krokach otrzymujemy DFT długości 2. W rozważanym przypadku ($N = 8$) kolejny krok wygląda następująco:

$$\frac{1}{N}\left(x(0) + x(2)\omega_4^k + x(4)\omega_4^{2k} + x(6)\omega_4^{3k}\right) =$$
$$= \frac{1}{N}\left(x(0) + x(4)\omega_4^{2k}\right) + \omega_4^k \frac{1}{N}\left(x(2) + x(6)\omega_4^{2k}\right) =$$
$$= \frac{1}{N}\left(x(0) + x(4)\omega_2^k\right) + \omega_4^k \frac{1}{N}\left(x(2) + x(6)\omega_2^k\right)$$

oraz

$$\frac{1}{N}\left(x(1)+x(3)\omega_4^k+x(5)\omega_4^{2k}+x(7)\omega_4^{3k}\right)=$$
$$=\frac{1}{N}\left(x(1)+x(5)\omega_4^{2k}\right)+\omega_4^k\frac{1}{N}\left(x(3)+x(7)\omega_4^{2k}\right)=$$
$$=\frac{1}{N}\left(x(1)+x(5)\omega_2^{k}\right)+\omega_4^k\frac{1}{N}\left(x(3)+x(7)\omega_2^{k}\right).$$

W każdym pojedynczym kroku wykonujemy operacje sumowania oraz mnożenia przez odpowiednią potęgę ω_N.

Złożoność czasowa algorytmu FFT

Zgodnie z powyższym równaniem FFT długości 8 jest zastąpiona dwiema FFT długości 4. Aby wyznaczyć wszystkie elementy FFT ($k = 0, \ldots, 7$) musimy do liczby operacji potrzebnych na wyzna-

czenie dwóch FFT długości 4 dodać 8 operacji polegających na dodaniu do aktualnej wartości iloczynu dwóch liczb zespolonych (tzn. a←a+b*c, zwanych operacjami MAC). Ten wynik może być uogólniony dla FFT długości N, gdzie N jest potęgą 2. Jeżeli przyjmiemy, że C_N jest liczbą MAC operacji w N-tym kroku, to $C_N = 2C_{N/2}+N$ oraz ostatecznie mamy

$$C_N = N \log_2(N).$$

Musimy jeszcze uwzględnić przenumerowanie elementów ciągu. Indeksy danych wejściowych względem wyjściowych (lub na odwrót) tworzone są zgodnie z zasadą odwrotnej kolejności bitów w indeksie. Poniższa tabela pokazuje ustawienie danych wejściowych i wyjściowych dla $N = 8$.

Stara numeracja	Zapis binarny	Odwrotna kolejność	Nowa numeracja
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Zadanie 1. *Napisz w MATLAB-ie program wyznaczający szybką transformatę Fouriera. Deklaracja implementowanej funkcji powinna być następująca:*

$$c=dfft(x),$$

gdzie
x *– ciąg wejściowy,*
c *– dyskretna transformata Fouriera.*

Rozwiązanie: $dfft.m$

□

Zadanie 2. *Przetestuj program* $dfft$ *dla przykładów z poprzedniego rozdziału.*