Пахомов С.Н., Просеков О.В.

ВЫЧИСЛИТЕЛЬНЫЕ АСПЕКТЫ БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ[1]

В области цифровой обработки сигналов большую роль играет дискретное преобразование Фурье (ДПФ). К настоящему времени разработан ряд эффективных схем его вычисления, в частности, алгоритм Кули-Тьюки быстрого преобразования Фурье (БПФ) [1, 2, 3]. Применение данного алгоритма позволяет существенно повысить быстродействие анализа спектра сигнала. Однако имеются некоторые возможности для его совершенствования. Последнее время необходимость в таком улучшении обострилась, поскольку зачастую приходится обрабатывать сигналы очень больших длин.

Данная работа посвящена двум вычислительным проблемам БПФ: нахождению перестановок и использованию комплексных экспонент. Получено рекуррентное соотношение, позволяющее эффективно вычислять перестановку reverse "на месте". Показано, как уменьшить количество необходимых комплексных констант, что очень важно при вычислении ДПФ больших размеров. В работе приводятся записи полученных результатов на алгоритмическом языке.

1. Введение

Пусть s — натуральное число и $N = 2^s$. Обозначим через $\mathbb{C}_N$ линейное пространство комплекснозначных N-периодических функций целочисленного аргумента $x = x(j)$, $j \in \mathbb{Z}$. Элементы пространства $\mathbb{C}_N$ будем называть *сигналами.* Дискретное преобразование Фурье $\mathcal{F}_N : \mathbb{C}_N \to \mathbb{C}_N$ сопоставляет сигналу

[1]Работа выполнена при финансовой поддержке РФФИ (номер проекта 01-01-00231).

x сигнал $X = \mathcal{F}_N(x)$ с компонентами

$$X(k) = \sum_{j=0}^{N-1} x(j)\omega_N^{-kj}, \quad k \in \mathbb{Z},$$

где $\omega_N = \exp(2\pi i/N)$. Сигнал X называется *спектром Фурье сигнала* x.

Рассмотрим один из вариантов быстрого вычисления ДПФ — алгоритм Кули-Тьюки быстрого преобразования Фурье *с прореживанием по времени* [1, 2, 3]. Для этого введем в рассмотрение перестановку rev_s, которая сопоставляет числу $j \in 0 : 2^s - 1$ число $\mathrm{rev}_s(j)$, двоичный код которого равен перевернутому s-разрядному двоичному коду числа j. Кроме того, положим $N_\nu = N/2^\nu$, $\Delta_\nu = 2^{\nu-1}$. Тогда алгоритм БПФ может быть записан в следующем виде:

$$x_0(j) = x\big(\mathrm{rev}_s(j)\big), \quad j \in 0 : N-1; \tag{1}$$

$$\begin{aligned} x_\nu(l + p\,\Delta_{\nu+1}) =& x_{\nu-1}(l + 2p\,\Delta_\nu) + \\ &+ \omega_{\Delta_{\nu+1}}^{-l} x_{\nu-1}\big(l + (2p+1)\Delta_\nu\big), \\ x_\nu(l + \Delta_\nu + p\,\Delta_{\nu+1}) =& x_{\nu-1}(l + 2p\,\Delta_\nu) - \\ &- \omega_{\Delta_{\nu+1}}^{-l} x_{\nu-1}\big(l + (2p+1)\Delta_\nu\big), \\ p \in 0 : N_\nu - 1, \quad & l \in 0 : \Delta_\nu - 1, \quad \nu \in 1 : s. \end{aligned} \tag{2}$$

Заметим, что индексы компонент в левой и правой частях формулы (2) совпадают, т.е. данная вычислительная схема является алгоритмом *с замещением*. Таким образом, в качестве $x_0, x_1, \ldots, x_s$ может быть использован один и тот же массив x.

Пусть процедура $reverse(x)$ производит вычисления по формуле (1) для сигнала x, функция $omega(l)$ вычисляет комплексные экспоненты $\omega_{\Delta_{\nu+1}}^{-l}$, процедура $butterfly(x1, x2, c)$ реализует "бабочку", т.е. выполняет следующие действия:

$$\begin{aligned} &m := x2 * c; \\ &x2 := x1 - m; \\ &x1 := x1 + m; \end{aligned}$$

Тогда алгоритм БПФ принимает следующий вид:

```
reverse(x);
for nu := 1 to s do
  for l := 0 to Delta_nu − 1 do
    begin
      c := omega(l);
      for p := 0 to N_nu − 1 do
        butterfly(x[l + 2 ∗ p ∗ Delta_nu],
                  x[l + (2 ∗ p + 1) ∗ Delta_nu], c);
    end;
```

Процедура $butterfly(x1, x2, c)$ не вызывает затруднений. Процедура $reverse(x)$ и функция $omega(l)$, в свою очередь, требуют детального рассмотрения.

2. Вычисление перестановок

Прежде всего, остановимся на проблеме вычисления перестановок $\mathrm{rev}_s(k)$, $k \in 0 : N-1$, используемых в формуле (1). Существует два подхода к решению данной задачи: предварительное и пошаговое вычисление перестановок.

Первый подход означает, что $\mathrm{rev}_s(k)$ для всех $k \in 0 : N-1$ определяется на некотором предварительном этапе, до непосредственно вычисления спектра. Достоинством этого метода является то, что во время работы алгоритма не тратится лишнее время на вычисление перестановок. Однако при таком подходе необходимо хранить в памяти заранее вычисленные элементы, что требует использования дополнительного массива длины N.

Остановимся на втором подходе — пошаговом вычислении перестановок, при котором $\mathrm{rev}_s(k)$ для всех $k \in 0 : N-1$ определяется непосредственно на этапе вычисления спектра. В отличие от предыдущего, этот метод не использует дополнительного массива. Однако при таком подходе особую роль играют временные характеристики, что вызывает необходимость в построении эффективного алгоритма вычисления перестановок.

Целое число $k \in 0 : N-1$ может быть единственным образом представлено в виде

$$k = \sum_{\nu=0}^{s-1} k_\nu 2^\nu, \quad k_\nu \in 0:1.$$

Запишем такое представление в форме $k = (k_{s-1}, k_{s-2}, \ldots, k_0)$. Тогда $\mathrm{rev}_s(k) = (k_0, k_1, \ldots, k_{s-1})$. Если число k отлично от $N-1$, то оно может быть записано в виде

$$k = (\underbrace{k_{s-1} \ldots k_{q+1}}_{s-q-1} 0 \underbrace{1 \ldots 1}_{q}), \quad q \in 0 : s-1. \tag{3}$$

Опишем алгоритм для пошагового вычисления перестановок. Для этого обозначим символом "$\oplus$" поразрядную логическую операцию "исключающее или".

Теорема 1. *Для любого* $k \in 0 : N-2$ *верно:*

$$\mathrm{rev}_s(k+1) = \mathrm{rev}_s(k) \oplus (N - \Delta_{s-q}). \tag{4}$$

Доказательство. Из представления (3) следует:

$$\begin{aligned} \mathrm{rev}_s(k) &= (\underbrace{1 \ldots 1}_{q} 0 \underbrace{k_{q+1} \ldots k_{s-1}}_{s-q-1}), \\ k+1 &= (\underbrace{k_{s-1} \ldots k_{q+1}}_{s-q-1} 1 \underbrace{0 \ldots 0}_{q}), \end{aligned} \tag{5}$$

В силу определения Δ_{s-q} можем записать:

$$\Delta_{s-q} = (\underbrace{0 \ldots 0}_{q} 1 \underbrace{0 \ldots 0}_{s-q-1}),$$

откуда

$$N - \Delta_{s-q} = (\underbrace{1 \ldots 1}_{q+1} \underbrace{0 \ldots 0}_{s-q-1}). \tag{6}$$

С учетом (5), (6) имеем:

$$\mathrm{rev}_s(k) \oplus (N - \Delta_{s-q}) = (\underbrace{0 \ldots 0}_{q} 1 \underbrace{k_{q+1} \ldots k_{s-1}}_{s-q-1}) = \mathrm{rev}_s(k+1).$$

$\square$

Формула (4) задает рекуррентное соотношение для вычисления перестановок, с учетом которого процедура $reverse(x)$ может быть записана в следующем виде:

$rev := 0;$
for $k := 1$ **to** $N-2$ **do**
  begin
    $Delta := N$ **shr** $1;$
    while rev **and** $Delta <> 0$ **do** $Delta := Delta$ **shr** $1;$
    $rev := rev$ **xor** $(N - Delta);$
    if $rev > k$ **then** $swap(x[k], x[rev]);$
  end;

Вычислим количество поразрядных логических операций и операций сдвигов в предложенном алгоритме.

Поразрядные логические операции используются в пятой и шестой строках алгоритма. Сначала рассмотрим цикл, стоящий в пятой строке. Для $N/2$ чисел, двоичная запись которых начинается с нуля, условие цикла проверяется один раз. Для $N/4$ чисел, первая цифра которых единица, а вторая — ноль, оно проверяется два раза и т.д. Число операций равно

$$L_1 = \sum_{i=1}^{s} \frac{N}{2^i} i = N \sum_{i=1}^{s} \left(\frac{i}{2^{i-1}} - \frac{i}{2^i} \right) = N \sum_{i=1}^{s} \frac{1}{2^{i-1}} - s = 2N - s - 2.$$

Число операций, выполняемых в шестой строке, очевидно, равно

$$L_2 = N - 2.$$

Общее число поразрядных логических операций определяется формулой

$$L = L_1 + L_2 = 3N - \log_2 N - 4.$$

Операции сдвигов используются в четвертой и пятой строках алгоритма. В четвертой строке их число равно

$$R_1 = N - 2.$$

Рассмотрим цикл, стоящий в пятой строке. Для $N/2$ чисел, двоичная запись которых начинается с нуля, он не выполняется ни разу. Для $N/4$ чисел, первая цифра которых единица, а вторая — ноль, он выполняется один раз и т.д. Число операций равно

$$R_2 = L_1 - (N-2) = N - s.$$

Общее число операций сдвигов определяется формулой

$$R = R_1 + R_2 = 2N - \log_2 N - 2.$$

Экспериментальные данные показывают, что БПФ с использованием полученного алгоритма по временным характеристикам не уступает БПФ с предварительным вычислением перестановок. Вместе с тем, отметим несомненную экономию памяти.

3. Вычисление комплексных экспонент

Остановимся на другой проблеме БПФ — вычислении комплексных экспонент $\omega_{\Delta_{\nu+1}}^{-l}$, $l \in 0 : \Delta_\nu - 1$, $\nu \in 1 : s$, используемых в формуле (2).

Введем обозначение

$$C_l = \cos\frac{2\pi l}{N}, \quad l \in 0 : \frac{N}{4}$$

и покажем, что указанных C_l достаточно для вычислений по формуле (2).

Сразу заметим, что при $\nu = 1$ значения C_l не требуются, поскольку в этом случае $l = 0$ и $\omega_{\Delta_{\nu+1}}^{-l} = 1$.

Теорема 2. *Для любого $\nu \in 2 : s$ верно:*

$$\omega_{\Delta_{\nu+1}}^{-l} = C_{N_\nu l} - i\, C_{N_2 - N_\nu l}, \quad l \in 0 : \Delta_{\nu-1} - 1; \tag{7}$$

$$\omega_{\Delta_{\nu+1}}^{-l} = -C_{N_1 - N_\nu l} - i\, C_{N_\nu l - N_2}, \quad l \in \Delta_{\nu-1} : \Delta_\nu - 1. \tag{8}$$

При этом индексы используемых C_l лежат в интервале $0 : N/4$.

Доказательство. Прежде всего, отметим, что

$$\omega_{\Delta_{\nu+1}}^{-l} = \omega_N^{-N_\nu l} = \cos\frac{2\pi N_\nu l}{N} - i\sin\frac{2\pi N_\nu l}{N}. \tag{9}$$

Используя тригонометрические формулы приведения, имеем:

$$\sin\frac{2\pi N_\nu l}{N} = \cos\frac{2\pi\left(\frac{N}{4} - N_\nu l\right)}{N} = C_{N_2 - N_\nu l} = C_{N_\nu l - N_2}; \tag{10}$$

$$\cos\frac{2\pi N_\nu l}{N} = -\cos\frac{2\pi\left(\frac{N}{2} - N_\nu l\right)}{N} = -C_{N_1 - N_\nu l}. \tag{11}$$

Формулы (7), (8) следуют из соотношения (9) с учетом (10) и (11).

Покажем, что индексы используемых C_l лежат в интервале $0 : N/4$. Действительно, при $l \in 0 : \Delta_{\nu-1} - 1$ можем записать:

$$\begin{gathered} N_\nu\, l = 0, N_\nu, \ldots, \frac{N}{4} - N_\nu; \\ N_2 - N_\nu\, l = \frac{N}{4}, \frac{N}{4} - N_\nu, \ldots, N_\nu. \end{gathered} \tag{12}$$

При $l \in \Delta_{\nu-1} : \Delta_\nu - 1$ имеем:

$$\begin{gathered} N_\nu l = \frac{N}{4}, \frac{N}{4} + N_\nu, \ldots, \frac{N}{2} - N_\nu; \\ N_1 - N_\nu l = \frac{N}{4}, \frac{N}{4} - N_\nu, \ldots, N_\nu; \\ N_\nu l - N_2 = 0, N_\nu, \ldots, \frac{N}{4} - N_\nu. \end{gathered} \tag{13}$$

$\square$

Отметим, что при $l = 0$ нет необходимости прибегать к соотношению (7), поскольку $\omega_{\Delta_{\nu+1}}^0 = 1$. При $l = \Delta_{\nu-1}$ можно избежать соотношения (8), т.к. в этом случае $\omega_{\Delta_{\nu+1}}^{-\Delta_{\nu-1}} = -i$. Таким образом, в силу (12), (13) индексы используемых C_l лежат в интервале $0 : N/4 - 1$.

С учетом вышесказанного функция $omega(l)$ может быть записана в следующем виде:

$Nl := N\_nu * l;$
case l **of**
  0 : **begin**
      $res.re := 1;$
      $res.im := 0;$
    end;
  1 .. $Delta\_nu$ **shr** $1 - 1$:
    begin
      $res.re := c[Nl];$
      $res.im := -c[N$ **shr** $2 - Nl];$
    end;
  $Delta\_nu$ **shr** 1 :
    begin
      $res.re := 0;$
      $res.im := -1;$
    end;
  $Delta\_nu$ **shr** $1 + 1$ .. $Delta\_nu - 1$:
    begin
      $res.re := -c[N$ **shr** $1 - Nl];$
      $res.im := -c[Nl - N$ **shr** $2];$
    end;
end;
$omega := res;$

Поскольку в алгоритме БПФ l пробегает значения от 0 до $Delta\_nu - 1$ последовательно, то в функции $omega(l)$ нет необходимости. Достаточно лишь в исходном алгоритме разбить цикл по l на 4 части: $l = 0$, $l \in 1 : \Delta_{\nu-1} - 1$, $l = \Delta_{\nu-1}$ и $l \in \Delta_{\nu-1} + 1 : \Delta_\nu$.

Учитывая, что C_l представляют собой вещественные числа, использование соотношений (7) и (8) для вычисления комплексных экспонент позволяет в 4 раза снизить объем потребляемой памяти.

ЛИТЕРАТУРА

1. *Cooley J., Tukey J.* An algorithm for the machine calculation of complex Fourier series// Math. Comput. 1965. Vol. 19. P. 297-301.

2. *Блейхут Р.* Быстрые алгоритмы цифровой обработки сигналов. Пер. с англ. М., 1989. 448 с.

3. *Малоземов В.Н., Третьяков А.А.* Алгоритм Кули-Тьюки и дискретное преобразование Хаара// Вестн. С.-Петербург. ун-та. Сер. 1. 1998. Вып. 3 (№15). С. 31-34.