УДК 519.6 : 681.3

ПАРАЛЛЕЛЬНЫЕ СХЕМЫ НЕКОТОРЫХ ДИСКРЕТНЫХ ОРТОГОНАЛЬНЫХ ПРЕОБРАЗОВАНИЙ*

Я. Е. Ромм, В. В. Забеглов

Государственное образовательное учреждение высшего профессионального образования «Таганрогский государственный педагогический институт», 347926, г. Таганрог, ул. Инициативная, 48
E-mail: romm@list.ru

Изложены параллельные видоизменения преобразований Уолша, Хаара с временной сложностью $O(\log_2 N)$, а также быстрого вейвлет- и пилообразного преобразований. Параллельное вычисление базиса дискретного преобразования Фурье выполняется с оценкой $O(1)$, параллельный алгоритм быстрого вейвлет-преобразования оценивается временной сложностью $O(\log_2 \log_2 N \times \log_2 N)$.

Ключевые слова: дискретные ортогональные преобразования, преобразования Уолша, Хаара, вейвлет-преобразование, параллельные схемы, временная сложность.

Введение. Ортогональные преобразования используются для цифровой обработки сигналов, представляющих сейсмические, акустические, биомедицинские данные, а также данные обработки изображений, речевых сигналов, анализа и проектирования систем связи и др. К наиболее часто применяемым относятся преобразования Фурье, Хаара, Уолша, вейвлет- и пилообразное преобразования. При этом актуальны задачи сокращения времени и объёма вычислений. В предлагаемой работе обсуждается повышение быстродействия на основе преобразования данных схем к параллельной форме. Параллельные алгоритмы рассматриваются на модели неветвящихся параллельных программ [1, 2] без учёта обмена. Временная сложность (или время) $T(R)$ измеряется количеством последовательных шагов алгоритма, при этом R — количество процессоров.

Параллельный алгоритм пилообразного преобразования. Пилообразное преобразование [3] можно записать в виде

$$D_N = S_N \times X_N, \tag{1}$$

где $D'_N = [D(0), D(1), \ldots, D(N-1)]$ — вектор коэффициентов; $X'_N = [X(0), X(1), \ldots$ $\ldots, X(N-1)]$ — вектор входной последовательности; S_N — матрица размера $N \times N$, которая задаётся рекуррентной формулой

$$S_N = \frac{1}{\sqrt{2}} \left[\begin{array}{cc:c:cc:c} 1 & 0 & & 1 & 0 & \\ a_n & b_n & 0 & -a_n & b_n & 0 \\ 0 & & I_{(N/2)-2} & 0 & & I_{(N/2)-2} \\ \hdashline 1 & 0 & & 0 & -1 & \\ -b_n & a_n & 0 & b_n & a_n & 0 \\ 0 & & I_{(N/2)-2} & 0 & & -I_{(N/2)-2} \end{array}\right], \tag{2}$$

*Работа выполнена при поддержке Российского фонда фундаментальных исследований (проект № 10-07-00178а от 2010 г.).

$$S_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad n = \log_2 N$$

(здесь I_m — единичная квадратная клетка $m \times m$). Постоянные a_k и b_k вычисляются по формулам [4]

$$a_k = \frac{2^{k-1}}{\sqrt{1 + \frac{4}{3}(2^{2(k-1)} - 1)}}, \quad b_k = \frac{a_k}{2a_{k-1}} = \sqrt{\frac{2^{2(k-1)} - 1}{2^{2k} - 1}}, \quad k = 1, 2, \ldots, \log_2 N. \tag{3}$$

Коэффициенты (3) для всех $k = 1, 2, \ldots, \log_2 N$ можно вычислить наперёд и записать в память компьютера. При ограничении на память используются схемы кусочно-полиномиальной аппроксимации [5], которые позволяют вычислять радикалы с временной сложностью $O(1)$; разновидность динамического вычисления значений (3) по схеме Стоуна с временной сложностью $O(\log_2(\log_2 N))$ на $\log_2 N$ процессорах приведена в [6]. Далее предполагается, что все значения (3) вычислены и априори заданы. По рекуррентности матрицу S_N из (2) представим в виде произведения $\log_2 N$ матриц, которое вычислим за логарифмическое от числа сомножителей количество шагов по схеме сдваивания, равное $O(\log_2 \log_2 N)$.

Обозначим левую матрицу в правой части (2) через P_N:

$$S_N = \frac{1}{\sqrt{2}} P_N \left[\begin{array}{c|c} S_{N/2} & 0 \\ \hline 0 & S_{N/2} \end{array}\right]. \tag{4}$$

Из (4) по рекуррентности

$$S_N = \frac{1}{\sqrt{4}} P_N \left[\begin{array}{c|c} P_{N/2} \left[\begin{array}{c|c} S_{N/4} & 0 \\ \hline 0 & S_{N/4} \end{array}\right] & 0 \\ \hline 0 & P_{N/2} \left[\begin{array}{c|c} S_{N/4} & 0 \\ \hline 0 & S_{N/4} \end{array}\right] \end{array}\right]. \tag{5}$$

С учётом клеточной структуры матриц соотношение (5) преобразуется к виду

$$S_N = \frac{1}{\sqrt{4}} P_N \left[\begin{array}{c|c} P_{N/2} & 0 \\ \hline 0 & P_{N/2} \end{array}\right] \left[\begin{array}{c|c} \begin{array}{c|c} S_{N/4} & 0 \\ \hline 0 & S_{N/4} \end{array} & 0 \\ \hline 0 & \begin{array}{c|c} S_{N/4} & 0 \\ \hline 0 & S_{N/4} \end{array} \end{array}\right]. \tag{6}$$

Аналогично выражается $S_{N/4}, S_{N/8}, \ldots, S_2$. Отсюда произведение (6) эквивалентно

произведению $\log_2 N$ матриц

$$S_N = \frac{1}{\sqrt{N}} P_N \left[\begin{array}{c:c} P_{N/2} & 0 \\ \hdashline 0 & P_{N/2} \end{array}\right] \cdots \begin{bmatrix} P_4 & 0 & 0 & 0 & 0 \\ 0 & P_4 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & P_4 & 0 \\ 0 & 0 & 0 & 0 & P_4 \end{bmatrix} \begin{bmatrix} S_2 & 0 & 0 & 0 & 0 \\ 0 & S_2 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & S_2 & 0 \\ 0 & 0 & 0 & 0 & S_2 \end{bmatrix}. \quad (7)$$

Если не учитывать клеточно-диагональную структуру матриц из (7), можно воспользоваться их умножением по схеме сдваивания. Умножение каждой пары матриц также выполняется с применением схемы сдваивания за $\log_2 N$ шагов. Правая часть (7) разбивается на $\dfrac{\log_2 N}{2}$ пар матриц, каждой паре для умножения требуется не больше N^3 процессоров, общее число процессоров $N^3 \dfrac{\log_2 N}{2}$. В этом случае временная сложность вычисления (7) составит

$$T\left(N^3 \frac{\log_2 N}{2}\right) = O(\log_2 N \times \log_2 \log_2 N). \quad (8)$$

Число процессоров в (8), а также время умножения существенно сокращаются, если учесть одинаковую клеточную структуру матриц в (7), клетки отличаются только номерами коэффициентов и размерностью. Пусть умножение матриц выполняется по клеткам. Каждая из них, в свою очередь, состоит из четырёх вложенных клеток, которые отличаются от единичной четырьмя элементами в левом верхнем углу. Отсюда

$$\begin{bmatrix} a & b & 0 & \cdots & 0 \\ c & d & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} e & f & 0 & \cdots & 0 \\ g & h & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} ae+bg & af+bh & 0 & \cdots & 0 \\ ce+dg & cf+dh & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Для данной операции требуется выполнить восемь умножений и четыре алгебраических сложения.

На первом шаге схемы сдваивания матриц из (7) в каждой паре вычисляется восемь произведений клеток: четыре клетки первой матрицы умножаются на две клетки второй. Первый шаг соответствует началу закономерности: $4 \times 2 = 8$. На втором шаге число произведений клеток составит $16 \times 4 = 8^2$, на третьем — $256 \times 16 = 8^4$, на шаге i — $8^{2^{i-1}}$, $i = 1, 2, \ldots, \log_2 \log_2 N$. На каждом шаге вдвое увеличивается показатель степени 8, в 2 раза уменьшается количество пар произведений, суммарное число произведений клеток во всех парах изменяется следующим образом:

$$8\frac{\log_2 N}{2},\ 8^2\frac{\log_2 N}{2^2},\ 8^4\frac{\log_2 N}{2^3},\ \ldots,\ 8^{2^{\log_2 \log_2 N - 1}} \frac{\log_2 N}{2^{\log_2 \log_2 N}} = N^{3/2}.$$

Число шагов в правой части (8) сокращается в $\log_2 N$ раз, так как умножение матрицы на матрицу свелось к взаимно независимому синхронному умножению клеток, определяемому восемью умножениями и четырьмя сложениями, за время $O(1)$.

Таким образом, имеет место

Лемма 1. Матрицу (2) по схеме (7) можно вычислить параллельно с временной сложностью

$$T(N^{3/2}) = O(\log_2 \log_2 N).$$

Нетрудно заметить, что не только элементы (7) вычисляются априори и хранятся в памяти, но это можно сделать для всего произведения матриц в правой части (7). В этом предположении имеет место

Лемма 2. Матрицу (2) можно определить за время $T(N^2) = O(1)$ посредством адресации к хранимым элементам.

В максимально параллельной форме само преобразование (1) выполнимо с помощью схемы сдваивания парных произведений. Как известно, временная сложность в этом случае составит

$$T(N^2) = O(\log_2 N). \tag{9}$$

Таким образом, имеет место

Теорема 1. Пилообразное преобразование, ограниченное N членами, в условиях леммы 1 можно выполнить с временной сложностью

$$T(N^2) = O(\log_2 N) + O(\log_2 \log_2 N),$$

в условиях леммы 2 данное преобразование выполняется с оценкой (9).

Число процессоров, указанное в лемме 1, поглощается максимально параллельной формой умножения матрицы на вектор, при этом улучшение оценки (9) в асимптотике не оправдывает избыточные затраты памяти, поскольку $O(\log_2 N) + O(\log_2 \log_2 N) = O(\log_2 N)$. Однако в теореме 1 отражены две схемы формирования матриц пилообразного преобразования, представленные в обеих леммах.

Параллельное вычисление преобразования Уолша. Для нормированных функций Уолша принято обозначение $\mathrm{wal}(r, \theta)$, где r — номер функции, а θ находится в полуинтервале $0 \le \theta < 1$. Рассматривается множество функций $\mathrm{wal}(r, \theta)$ при $r = 0, 1, \ldots, N-1$, где $N = 2^m$ и $m = 1, 2, 3, \ldots$. Дискретные функции Уолша получаются из непрерывных при отсчёте значений последних в соответствующих точках полуинтервала. Выбирается N точек с равномерным шагом длиной $1/N$. Матрица Уолша $H_w(N)$ состоит из элементов $h_{rv}^{(w)}$, которые представляют собой дискретные значения функций Уолша $\mathrm{wal}(r, \theta)$, где $v = 0, 1, \ldots, N-1$ — номер отсчёта. Преобразование Уолша определяется как произведение вида

$$B(N) = \frac{1}{N} H_w(N) X(N), \tag{10}$$

где $X(N)$ — транспонированный вектор входной последовательности $X'_N = [X(0), X(1), \ldots$ $\ldots, X(N-1)]$.

Функции Уолша можно выразить аналитически для любого N в виде следующего соотношения [7]:

$$\mathrm{wal}(r, \theta) = \prod_{j=1}^{\log_2 N} \mathrm{rad}(j, \theta)^{r_{(\log_2 N - j + 1)} \oplus r_{(\log_2 N - j)}}, \tag{11}$$

где r_j — j-й разряд представления числа r в двоичной системе счисления; $\oplus$ — символ поразрядного суммирования по модулю 2; $\mathrm{rad}(j,\theta)$ — функции Радемахера, определяемые соотношением

$$\mathrm{rad}(j,\theta) = \mathrm{sign}(\sin(2^j\pi,\theta)), \quad \theta \in [0,\ 1). \tag{12}$$

В (12) j — целое неотрицательное число; $\mathrm{sign}(x)$ — функция, такая что $\mathrm{sign}(x) = 1$, если $x > 0$, и $\mathrm{sign}(x) = -1$, если $x < 0$; в точках разрыва значение функции (12) определяется по непрерывности справа.

Алгоритм построения матрицы $H_w(N)$ состоит из следующих этапов:

1. Вычисляются функции

$$f_j(\theta) = \sin(2^j\pi\theta), \quad \theta \in [0,\ 1),\ j = 1, 2, \ldots, \log_2 N. \tag{13}$$

2. Согласно (12) вычисляются функции Радемахера

$$\mathrm{rad}(j,\theta), \quad \theta \in [0,\ 1),\ j = 1, 2, \ldots, \log_2 N.$$

3. В соответствии с (11) вычисляются функции Уолша

$$\mathrm{wal}(r,\theta), \quad \theta \in [0,\ 1),\ r = 0, 1, \ldots, N-1,\ N = 2^m,\ m = 1, 2, \ldots\ .$$

Далее приводятся схемы параллельного вычисления матрицы $H_w(N)$ и преобразования (10).

Для вычисления функций (13) используется схема кусочно-полиномиальной аппроксимации функций на основе интерполяционного полинома Ньютона с минимизацией временной сложности до значения порядка $O(1)$ [5]. Выбирается система непересекающихся подынтервалов равной длины, объединение которых совпадает с

$$[0,\ 1) = \bigcup_{i=0}^{P-1} [\theta_i, \theta_{i+1}). \tag{14}$$

Для определённости P предполагается целой степенью по основанию 2, поэтому

$$\theta_{i+1} - \theta_i = 1/P, \quad i = 0, 1, \ldots, P-1,\ P = 2^k,\ k = 0, 1, \ldots\ . \tag{15}$$

При априори заданной границе ε абсолютной погрешности аппроксимации функций (13) для каждого подынтервала из (14), (15) строится интерполяционный полином Ньютона степени n с числовыми коэффициентами

$$P_n(\theta) = a_{nij}\theta^n + \ldots + a_{2ij}\theta^2 + a_{1ij}\theta + a_{0ij}, \quad \theta \in [\theta_i, \theta_{i+1}),\ n = \mathrm{const},\ i = 0, 1, \ldots, P-1, \tag{16}$$

где i — номер подынтервала, j — индекс функции из (13). Для преобразования интерполяционного полинома Ньютона к форме (16) используется алгоритм восстановления коэффициентов полинома по его корням, инвариантный относительно номера коэффициента [8]. Одинаковая для всех номеров подынтервалов степень n выбирается минимальной для достижения заданной точности

$$\left| f_j(\theta) - P_n(\theta) \right| \le \varepsilon, \quad \theta \in [\theta_i, \theta_{i+1}),\ i = 0, 1, \ldots, P-1. \tag{17}$$

С этой целью условие (17) проверяется для наименьшего k из (15) ($k = 0$) в равноотстоящих проверочных точках с шагом, существенно меньшим расстояния между узлами интерполяции. При нарушении (17) хотя бы на одном проверочном шаге хоть одного

подынтервала значение n увеличивается на 1 при сохранении k; это продолжается до нарушения допустимых границ значений n; при их нарушении снова делается переход к минимальному n, но уже при удвоенном количестве подынтервалов ($k = 1$) и т. д. В качестве искомого фиксируется минимальное n при наименьшем количестве подынтервалов. Такое n определяется для каждой функции из (13) и каждого подынтервала (14), (15); набор коэффициентов (16) записывается в память компьютера в виде массива элементов с индексами i, j. Затем, чтобы вычислить функцию с номером j, дешифруется номер i; номера i и j служат адресом выборки коэффициентов (16). Если $\theta \in [\theta_i, \theta_{i+1})$, то $i = \mathrm{int}(\theta/h)$, где int — целая часть числа, $h = \theta_{i+1} - \theta_i$. Время данной дешифрации

$$t = O(1). \tag{18}$$

С учётом (18) время вычисления функции фактически зависит только от n из (16). По схеме Горнера значение полинома (16) находится за время $t(1) = n(t_y + t_c)$, где t_y, t_c — времена бинарного умножения и сложения соответственно. Поскольку n фиксировано, время вычисления функции сохраняет порядок (18). В частности, для $n = 1$ вычисление функции выполняется с точностью до 10^{-8} при $k = 14$; для $n = 2$ — с точностью до 10^{-14} при $k = 15$; для $n = 3$ — с точностью до 10^{-18} при $k = 14$ [5].

Помимо функций (13) данным способом далее определяются элементы базиса преобразования Фурье.

В результате считывания коэффициентов и вычисления полиномиального приближения функций (13) получаем систему функций Радемахера (12). Для нахождения всех значений (12) за время (18) потребуется следующее количество процессоров. Каждому отсчёту с номером ℓ с целью одновременного вычисления всех функций $f_j(\theta)$ из (13) требуется поставить в соответствие $\log_2 N$ процессоров. Эти процессоры определят знак функций (12) одновременно для всех $j = 1, 2, \ldots, \log_2 N$, где j взято из (13), при этом $\ell = 0, 1, \ldots, N-1$. Отсюда количество процессоров $R = N \log_2 N$.

Из изложенного вытекает

Лемма 3. Систему функций Радемахера (12) при помощи кусочно-полиномиальной схемы можно вычислить параллельно с временной сложностью

$$T(N \log_2 N) = O(1). \tag{19}$$

Иначе, функции (13) и соответственно (12) вычисляются с помощью полиномов Чебышёва I-го и II-го рода $T_\ell(x)$ и $U_\ell(x)$ на основе известных рекуррентных соотношений [9]:

$$\left.\begin{aligned} &T_{\ell+1}(x_i) = 2x_i T_\ell(x_i) - T_{\ell-1}(x_i), \\ &T_0(x_i) = 1, \;\; T_1(x_i) = x_i, \end{aligned}\right\} \tag{20}$$

$$\left.\begin{aligned} &U_{\ell+1}(x_i) = 2x_i U_\ell(x_i) - U_{\ell-1}(x_i), \\ &U_0(x_i) = 1, \;\; U_1(x_i) = 2x_i. \end{aligned}\right\} \tag{21}$$

Зависимости (20), (21) представим в матричной форме [9]:

$$\begin{pmatrix} T_{\ell+1}(x_i) \\ T_\ell(x_i) \end{pmatrix} = \begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} T_\ell(x_i) \\ T_{\ell-1}(x_i) \end{pmatrix},$$

$$\begin{pmatrix} U_{\ell+1}(x_i) \\ U_\ell(x_i) \end{pmatrix} = \begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} U_\ell(x_i) \\ U_{\ell-1}(x_i) \end{pmatrix}, \quad \ell = 1, 2, \ldots,$$

с такими же начальными значениями. Отсюда

$$\begin{pmatrix} T_{\ell+1}(x_i) \\ T_\ell(x_i) \end{pmatrix} = \begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix}^\ell \begin{pmatrix} T_1(x_i) \\ T_0(x_i) \end{pmatrix}, \quad \begin{pmatrix} U_{\ell+1}(x_i) \\ U_\ell(x_i) \end{pmatrix} = \begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix}^\ell \begin{pmatrix} U_1(x_i) \\ U_0(x_i) \end{pmatrix}.$$

С помощью схемы Стоуна [10] вычисление $\begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix}^\ell$ для всех значений ℓ, $\ell = 1, 2, \ldots, N$, выполняется на $O(N^2)$ процессорах с временной сложностью $O(\log_2 N)$. Первоначально находятся все двоичные степени:

$$A = \begin{pmatrix} 2x_i & -1 \\ 1 & 0 \end{pmatrix}, \quad -A^{2^0} = A, \ldots, A^{2^m} = (A^{2^{m-1}})^2, \quad m = 1, \ldots, \log_2 N.$$

Пусть ℓ имеет двоичное разложение

$$\ell = \sum_{m=0}^{\log_2 N - 1} \alpha_m 2^m, \quad \alpha_m = \begin{cases} 0, \\ 1, \end{cases} \quad \ell = 1, 2, \ldots, N. \tag{22}$$

При каждом x_i каждому ℓ из (22) взаимно однозначно сопоставляются процессоры с индексами ℓ, i, которые параллельно найдут все степени матрицы с показателем ℓ для всех $i = 0, 1, \ldots, N-1$. При этом ℓ-й процессор, $\ell = 1, 2, \ldots, N$, перемножает те двоичные степени матрицы, перед показателем 2^m которой в (22) $\alpha_m = 1$. Отсюда временная сложность данной схемы $T(N^2) = O(\log_2 N)$.

Используя известные соотношения [9], $\sin(\ell\pi\theta_i)$, $\ell = 1, 2, \ldots, N$, выразим через полиномы Чебышёва:

$$\sin((2\ell+1)\pi\theta_i) = (-1)^\ell T_{2\ell+1}(\sin(\pi\theta_i)), \quad \sin(2\ell\pi\theta_i) = (-1)^{\ell-1}\cos(\pi\theta_i) U_{2\ell-1}(\sin(\pi\theta_i)),$$

$$\cos((2\ell+1)\pi\theta_i) = (-1)^\ell \cos(\pi\theta_i) U_{2\ell}(\sin(\pi\theta_i)), \quad \cos(2\ell\pi\theta_i) = (-1)^\ell T_{2\ell}(\sin(\pi\theta_i)).$$

С учётом (12), (13) вычисляется система функций Радемахера.

Таким образом, имеет место

Лемма 4. С помощью схемы Стоуна систему функций Радемахера (12) можно вычислить параллельно с временной сложностью

$$T(N^2) = O(\log_2 N). \tag{23}$$

Оценка (23) уступает (19), но алгоритм не требует затрат постоянной памяти.

Функции Радемахера строятся без вычисления синусов из (13) с установкой только знака их значений для каждого θ_i, $i = 0, 1, \ldots, N-1$. С этой целью воспользуемся формулой редукции [11, 12]

$$\left.\begin{aligned} &\sin(x) = (-1)^K \operatorname{sign}(x) \sin(\pi z), \\ &K = \operatorname{int}\Big(\frac{|x|}{\pi}\Big), \quad z = \Big\{\frac{|x|}{\pi}\Big\}, \end{aligned}\right\} \tag{24}$$

где $\{\alpha\}$ дробная и $\mathrm{int}(\alpha)$ — целая части числа α, с учётом (12) $x = 2^j\pi\theta_i$. Для определения знака синусов в (13) достаточно установить значение первого множителя $(-1)^K$ в формуле (24), оставшиеся два множителя ($\mathrm{sign}(x)$ и $\sin(\pi z)$) при рассматриваемых x и z не принимают отрицательных значений. Отсюда функции Радемахера (12) находятся с помощью формулы

$$\mathrm{rad}(j,\theta_i) = (-1)^{\mathrm{int}(2^j\theta_i)}, \quad \theta_i \in [0,\,1),\ j = 1,2,\ldots,\log_2 N,\ i = 0,1,\ldots,N-1. \tag{25}$$

Параллельный алгоритм строится следующим образом. Все степени двойки 2^j в (25) предполагаются априори вычисленными и хранимыми в памяти компьютера. При каждом j каждому отсчёту θ_i ставится в соответствие процессор с индексами i, j. Процессоры параллельно по всем номерам i и j выполнят умножения $2^j\theta_i$ и определят $\mathrm{int}(2^j\theta_i)$, затем по значению, например, младшего разряда целого двоичного числа определят чётность, которой соответствует искомый знак в (25).

С учётом границ индексов в (25) из изложенного следует

Лемма 5. На основе (24), (25) систему функций Радемахера (12) можно вычислить параллельно с временной сложностью

$$T(N\log_2 N) = O(1). \tag{26}$$

Оценка (26) аналогично (19) достигает минимальной временной сложности ценой избыточных затрат памяти.

По значениям функций Радемахера (12) в точках $\theta = \theta_i$, $i = 0,1,\ldots,N-1$, $\theta_i - \theta_{i-1} = \mathrm{const}$ согласно правилу (11) вычисляются функции Уолша. Каждой такой функции с номером r, $r = 0,1,\ldots,N-1$, ставится в соответствие $N\log_2 N$ процессоров для того, чтобы произведение в (11) при каждом $\theta = \theta_i$ вычислялось по схеме сдваивания на $\mathrm{int}\left(\frac{1}{2}\log_2 N\right)$ процессорах. Работая параллельно по всем рассматриваемым r и i, процессоры перемножат функции Радемахера с временной сложностью

$$T(N^2\log_2 N) = O(\log_2\log_2 N). \tag{27}$$

Согласно изложенному с учётом $O(\log_2\log_2 N) + O(1) = O(\log_2\log_2 N)$ имеет место

Лемма 6. В условиях леммы 4 матрицу преобразования Уолша размера $N \times N$ можно параллельно вычислить с временной сложностью

$$T(N^2\log_2 N) = O(\log_2 N) + O(\log_2\log_2 N),$$

в условиях лемм 3 и 5 — с временной сложностью (27).

Замечание 1. Оценка (27) может быть улучшена до значения $O(1)$ с сокращением числа процессоров в $O(\log_2 N)$, если вместо схемы сдваивания в (11) применить битовый счётчик числа отрицательных двоичных единиц (сомножителей).

Замечание 2. Ещё одна параллельная схема получится, если матрицу преобразования Уолша $H_w(N)$ вычислять непосредственно поэлементно. Пусть r_i и v_i — цифры i-го разряда в двоичном представлении целых чисел r и v соответственно:

$$r = (r_{\log_2 N-1}\ r_{\log_2 N-2}\ \ldots\ r_1\ r_0)_2, \quad v = (v_{\log_2 N-1}\ v_{\log_2 N-2}\ \ldots\ v_1\ v_0)_2.$$

Элементы $h_{rv}^{(w)}$ матрицы Уолша имеют вид [3]

$$h_{rv}^{(w)} = (-1)^{\sum\limits_{i=0}^{\log_2 N-1} u_i(r)v_i}, \quad r = 0,1,\ldots,N-1,\ v = 0,1,\ldots,N-1, \tag{28}$$

где $u_0(r) = r_{\log_2 N-1}$; $u_1(r) = r_{\log_2 N-1} + r_{\log_2 N-2}$; $u_2(r) = r_{\log_2 N-2} + r_{\log_2 N-3}, \ldots$ $\ldots, u_{\log_2 N-1}(r) = r_1 + r_0$.

Все элементы матрицы вычисляются параллельно, причём суммы в показателях степени (28) находятся по схеме сдваивания. Временная сложность данного варианта совпадёт с (27) [13].

После вычисления всех элементов матрицы $H_w(N)$ находится само преобразование (10). Схема основана на естественном параллелизме и аналогична схеме пилообразного преобразования. Отсюда с учётом логарифмической оценки умножения матрицы на вектор из леммы 6 вытекает

Теорема 2. Преобразование Уолша можно выполнить параллельно с временной сложностью

$$T(N^2 \log_2 N) = O(\log_2 N),$$

при этом в условиях лемм 3 и 5 матрицу преобразования $H_w(N)$ вычисляют параллельно с оценкой (27).

С использованием схемы из замечания 2 получится аналогичная оценка, в условиях замечания 1 число процессоров сократилось бы до $O(N^2)$.

Замечание 3. В случае фиксированного N при каждом значении i в показателе степени (28) априори известны номер строки r_i и номер столбца v_i матрицы $H_w(N)$, поэтому априори можно вычислить их двоичное разложение, парные произведения $u_i(r)v_i$, а также сумму, задающую показатель. Отсюда все элементы $h_{rv}^{(w)}$ находятся априори и записываются в память компьютера. В этом случае время определения матрицы равно времени считывания $O(1)$. Для переменного N оценки $O(1)$ можно добиться на той же основе, применив счётчик одноразрядных единиц для вычисления суммы в показателе (28).

Параллельное выполнение преобразования Хаара. Функции Хаара определяются при каждом значении θ, $0 \le \theta < 1$, индексами j и r. Первый из них является номером подразделения в системе функций Хаара, второй — номером функции в соответствующем подразделении. Участки, на которых значения функций отличны от нуля, различны для разных подразделений. Для формирования $N = 2^m$ функций Хаара используется соотношение [3]

$$\mathrm{har}(0,\,0,\,\theta) = 1; \quad \mathrm{har}(j,\,r,\,\theta) = \begin{cases} 2^{j/2}, & \theta \in \left[\dfrac{r-1}{2^j},\, \dfrac{r-1/2}{2^j}\right), \\ -2^{j/2}, & \theta \in \left[\dfrac{r-1/2}{2^j},\, \dfrac{r}{2^j}\right), \\ 0, & \theta \notin \left[\dfrac{r-1}{2^j},\, \dfrac{r}{2^j}\right), \end{cases} \tag{29}$$

где $0 \le j < \log_2 N$, $1 \le r \le 2^j$.

Дискретная функция Хаара представляет собой последовательность значений (29) в точках $\theta = \theta_i$ с шагом $h = 1/N$ при $i = 0, 1, \ldots, N-1$. Множество дискретных функций Хаара $\mathrm{har}(j, r, \theta_i)$ образует матрицу преобразования Хаара $H^*(N)$ [3], где каждый номер подразделения j соответствует клетке из N столбцов отсчётов и 2^j строк — дискретных функций $\mathrm{har}(j, r, \theta_i)$, $0 \le j < \log_2 N$, $1 \le r \le 2^j$. Преобразование Хаара записывается в виде [3]

$$Y(N) = \frac{1}{N} H^*(N) X(N), \tag{30}$$

где $X(N)$ — вектор-столбец входных данных; $N = 2^m$, m — фиксированное натуральное число. В частности, при $m = 3$

$$H^*(8) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{pmatrix}.$$

В предположении, что все степени $2^{j/2}$ и $-2^{j/2}$ при $0 \le j < \log_2 N$ вычислены априори и записаны в память компьютера, матрица $H^*(N)$ строится параллельно следующим образом. Первая строка матрицы заполняется единицами. Далее, каждому ненулевому элементу матрицы ставится в соответствие процессор, который считывает этот элемент из памяти. Элементом матрицы с индексами $(2^j + n,\ 2^{m-j}(n-1) + k)$ при $1 \le n \le 2^j$, $1 \le k \le 2^{m-j-1}$ будет значение $2^{j/2}$, при $1 \le n \le 2^j$, $2^{m-j-1} + 1 \le k \le 2^{m-j}$ — значение $-2^{j/2}$, все остальные элементы — нули. Количество процессоров равно числу ненулевых элементов. Первая строка матрицы состоит из единиц, последовательно сверху вниз располагаются прямоугольные клетки с номерами j, $j = 0, 1, \ldots, \log_2 N$, размерность j-й из которых $2^j \times N$, причём ненулевые элементы в такой клетке располагаются ступенчато, так что их число всегда равно числу столбцов N. Умножением N на число клеток с учётом первой строки получится число ненулевых элементов матрицы, равное $N(\log_2 N + 1)$.

Данный способ не требует вычислений, поэтому временная сложность заполнения матрицы составит

$$T(R) = O(1), \quad R = N(\log_2 N + 1).$$

Преобразование (30) вычисляется с той спецификой, что при умножении строки на столбец множитель выносится за скобки, по схеме сдваивания выполняется алгебраическое сложение в скобках, затем — одно бинарное умножение. Вместо процессоров для данной операции целесообразно взять один умножитель и соответственное число сумматоров. В результате число умножителей равно числу строк матрицы без одной, а количество сумматоров по строкам меняется с учётом схемы сдваивания, как $N/2$ в первой строке и как $(1/2)\times(N/2^j)$ далее, без изменения внутри клетки с 2^j строками, где $j = 0, 1, \ldots, \log_2 N - 1$. Отсюда число сумматоров

$$\frac{N}{2} + \frac{1}{2} \sum_{j=0}^{\log_2 N - 1} N = \frac{N}{2}(1 + \log_2 N).$$

Без учёта произведений, в которые входят нули, потребуется $N(\log_2 N + 1)$ процессорных элементов, временная сложность схемы составит $T(N \log_2 N) = O(\log_2 N)$.

Отсюда вытекает

Теорема 3. Преобразование Хаара, ограниченное N членами, можно выполнить параллельно с временной сложностью $T(R) = O(\log_2 N)$, где количество процессоров $R = N(\log_2 N + 1)$ требуется при параллельном считывании из памяти всех элементов матрицы преобразования за время $O(1)$. Собственно преобразование (30) выполнимо с оценкой $T(R_1 + R_2) = t_y + (\log_2 N)t_c$, где t_y и t_c — времена бинарного умножения и сложения на $R_1 = \dfrac{N}{2}(1 + \log_2 N)$ сумматорах и $R_2 = N - 1$ умножителях соответственно.

Параллельное вычисление быстрого вейвлет-преобразования. Быстрое вейвлет-преобразование (БВП) [14] позволяет находить коэффициенты вейвлет-разложения с помощью алгебраических операций на основе свёртки по рекуррентным соотношениям

$$a_{j-1,\,k} = \sqrt{2}\sum_n h_n a_{j,\,n+2k}, \tag{31}$$

$$d_{j-1,\,k} = \sqrt{2}\sum_n g_n a_{j,\,n+2k}, \tag{32}$$

где $\{h_n\}$ — масштабирующий фильтр масштабирующей функции; $\{g_n\}$ — фильтр вейвлета; $\{a_{j,\,k}\}$ — начальные коэффициенты разложения функции $f(x)$ для некоторого уровня разрешения j, которые предполагаются известными. Далее рассматриваются вейвлеты Добеши, в которых масштабирующий фильтр $\{h_n\}$ и фильтр вейвлетов $\{g_n\}$ состоят из конечного числа ненулевых элементов.

Зафиксировав конкретное разрешение $j = j_0$, формулы вычисления коэффициентов (31), (32) можно представить в матричной форме [15]. На первом шаге БВП

$$\begin{bmatrix} a_{j_0-1,\,1} \\ a_{j_0-1,\,2} \\ \vdots \\ a_{j_0-1,\,N/2} \\ d_{j_0-1,\,1} \\ d_{j_0-1,\,2} \\ \vdots \\ d_{j_0-1,\,N/2} \end{bmatrix} = \sqrt{2} \underbrace{\begin{bmatrix} h_1 & h_2 & h_3 & h_4 & \cdots & h_L & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & h_1 & h_2 & h_3 & h_4 & \cdots & h_L & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_3 & h_4 & \cdots & h_L & 0 & \cdots & \cdots & 0 & 0 & 0 & h_1 & h_2 \\ g_1 & g_2 & g_3 & g_4 & \cdots & g_L & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & g_1 & g_2 & g_3 & g_4 & \cdots & g_L & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_3 & g_4 & \cdots & g_L & 0 & \cdots & \cdots & 0 & 0 & 0 & g_1 & g_2 \end{bmatrix}}_{N} \begin{bmatrix} a_{j_0,\,1} \\ a_{j_0,\,2} \\ \vdots \\ a_{j_0,\,N/2} \\ a_{j_0,\,N/2+1} \\ a_{j_0,\,N/2+2} \\ \vdots \\ a_{j_0,\,N} \end{bmatrix}, \tag{33}$$

где N — целая степень по основанию 2. В (33) матрица фильтров (МФ) имеет размер $N \times N$. Первая строка МФ, начиная с первого элемента, заполнена коэффициентами последовательности h_n, $n = 1, 2, \ldots, L$, остальные коэффициенты в этой строке нулевые. В каждой следующей строке коэффициенты h_n, $n = 1, 2, \ldots, L$, циклически сдвинуты на два номера вправо, при этом коэффициенты, выходящие за пределы матрицы справа, помещены в ту же строку слева без изменения нумерации. Коэффициентами h_n, $n = 1, 2, \ldots, L$, заполняются строки с номерами от $i = 1$ до $i = N/2$. Оставшиеся строки, начиная с номера

$N/2+1$, аналогично заполняются элементами g_n, $n=1,2,\ldots,L$. На втором шаге БВП

$$\begin{bmatrix} a_{j_0-2,\,1} \\ a_{j_0-2,\,2} \\ \vdots \\ a_{j_0-2,\,N/4} \\ d_{j_0-2,\,1} \\ d_{j_0-2,\,2} \\ \vdots \\ d_{j_0-2,\,N/4} \end{bmatrix} = \sqrt{2} \underbrace{\begin{bmatrix} h_1 & h_2 & h_3 & h_4 & \cdots & h_L & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & h_1 & h_2 & h_3 & h_4 & \cdots & h_L & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_3 & h_4 & \cdots & h_L & 0 & \cdots & \cdots & 0 & 0 & 0 & h_1 & h_2 \\ g_1 & g_2 & g_3 & g_4 & \cdots & g_L & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & g_1 & g_2 & g_3 & g_4 & \cdots & g_L & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_3 & g_4 & \cdots & g_L & 0 & \cdots & \cdots & 0 & 0 & 0 & g_1 & g_2 \end{bmatrix}}_{N/2} \begin{bmatrix} a_{j_0-1,\,1} \\ a_{j_0-1,\,2} \\ \vdots \\ a_{j_0-1,\,N/4} \\ a_{j_0-1,\,N/4+1} \\ a_{j_0-1,\,N/4+2} \\ \vdots \\ a_{j_0-1,\,N/2} \end{bmatrix}, \quad (34)$$

преобразованию подлежат элементы $a_{j_0-1,\,i}$, $i=1,2,\ldots,N/2$. МФ в (34) имеет вдвое меньший порядок, $(N/2)\times(N/2)$, в остальном строится по схеме первого шага.

Полное БВП заключается в вычислении наборов коэффициентов $d_{j_0-\ell,\,k}$, которые итеративно определяются через $a_{j_0-\ell+1,\,k}$, пока размер МФ будет не меньше длины фильтра, т. е. $N/2^\ell \geq L$.

Далее вектор коэффициентов $a_{j_0,\,k}$, $k=1,2,\ldots,N$, в правой части (33) обозначается $[A_0]$, вектор коэффициентов $a_{j_0-1,\,k}$ и $d_{j_0-1,\,k}$, $k=1,2,\ldots,N/2$, в левой части (33) — $\begin{bmatrix} A_1 \\ D_1 \end{bmatrix}$, МФ из (33) — $[M_0]$. Тогда (33) примет вид

$$\begin{bmatrix} A_1 \\ D_1 \end{bmatrix} = \sqrt{2}[M_0][A_0]. \quad (35)$$

Аналогично левая часть (34) обозначается $\begin{bmatrix} A_2 \\ D_2 \end{bmatrix}$, МФ из (34) — $[M_1]$, тогда (34) примет вид

$$\begin{bmatrix} A_2 \\ D_2 \end{bmatrix} = \sqrt{2}[M_1][A_1]. \quad (36)$$

Соотношения (35) и (36) являются двумя последовательными шагами вейвлет-преобразования, на выходе которых вычисляются два набора вейвлет-коэффициентов D_1 и D_2 длинами $N/2$ и $N/4$ соответственно. После вычисления D_1 из (35) преобразуется верхняя половина вектора $\begin{bmatrix} A_1 \\ D_1 \end{bmatrix}$, т. е. набор коэффициентов A_1. С использованием клеточной структуры матриц два первых шага БВП можно представить в виде соотношения

$$\begin{bmatrix} A_2 \\ D_2 \\ D_1 \end{bmatrix} = \sqrt{4} \left[\begin{array}{c|c} M_1 & 0 \\ \hline 0 & 1 \end{array}\right] [M_0][A_0],$$

которое проверяется непосредственно. По индукции вычисление вейвлет-коэффициентов

D_ℓ, $\ell = 1, 2, \ldots, \ell_0$, можно выразить произведением

$$\begin{bmatrix} A_{\ell_0} \\ D_{\ell_0} \\ \vdots \\ D_2 \\ D_1 \end{bmatrix} = 2^{\ell_0/2} \left[\begin{array}{cc|c|c} M_{\ell_0-1} & 0 & & \\ \hline 0 & 1 & 0 & \\ & & \ddots & 0 \\ \hline & 0 & & 1 \\ \hline & 0 & & 1 \end{array}\right] \cdots \left[\begin{array}{cc|c} M_2 & 0 & \\ \hline 0 & 1 & 0 \\ \hline & 0 & 1 \end{array}\right] \left[\begin{array}{cc} M_1 & 0 \\ \hline 0 & 1 \end{array}\right] [M_0][A_0], \quad (37)$$

где клетка M_ℓ имеет размерность $(N/2^\ell) \times (N/2^\ell)$, $\ell = 0, 1, \ldots, \ell_0 - 1$, все перемножаемые матрицы из (37) имеют размер $N \times N$.

Алгоритм выполнения БВП условно делится на две части: первая — вычисление произведения матриц, обозначаемого $[P]$, из правой части (37); вторая — вычисление вейвлет-коэффициентов

$$\begin{bmatrix} A_{\ell_0} \\ D_{\ell_0} \\ \vdots \\ D_2 \\ D_1 \end{bmatrix} = [P][A_0]. \quad (38)$$

Произведение $[P]$ находится по схеме сдваивания матриц в правой части (37) на основе естественного параллелизма с учётом клеточной структуры матриц, но внутренняя структура клеток не учитывается. В частности, не учитывается наличие нулей внутри клеток: временная сложность оценивается сверху, как если бы клетки M_ℓ, $\ell = 0, 1, \ldots, \ell_0 - 1$, были целиком заполнены.

Умножение первой пары матриц (согласно (37) нумерация производится справа налево) можно условно представить в виде

$$\begin{bmatrix} F & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & Q \\ T & G \end{bmatrix} = \begin{bmatrix} F \times R & F \times Q \\ T & G \end{bmatrix},$$

где все клетки имеют размерность $(N/2) \times (N/2)$. Отсюда временная сложность параллельного умножения рассматриваемых матриц составит

$$T\Big(2\Big(\frac{N}{2}\Big)^3\Big) = O\Big(\log_2 \frac{N}{2}\Big). \quad (39)$$

Умножение остальных пар матриц в (37) выполняется аналогично, при этом клетки отличаются размерностью. Временная сложность умножения второй справа налево пары матриц составит

$$T\Big(2\Big(\frac{N}{2^3}\Big)^3\Big) = O\Big(\log_2 \frac{N}{2^3}\Big).$$

По индукции для умножения ℓ-й справа налево пары матриц имеет место оценка

$$T\Big(2\Big(\frac{N}{2^{2\ell-1}}\Big)^3\Big) = O\Big(\log_2 \frac{N}{2^{2\ell-1}}\Big), \quad \ell = 1, \ldots, \mathrm{int}\Big(\frac{\ell_0}{2}\Big).$$

В результате временная сложность первого шага схемы сдваивания определяется правой частью (39) при количестве процессоров

$$R_1 = 2\Big(\frac{N}{2}\Big)^3 + 2\Big(\frac{N}{2^3}\Big)^3 + 2\Big(\frac{N}{2^5}\Big)^3 + \ldots + 2\Big(\frac{N}{2^{\ell_0-3}}\Big)^3 + 2\Big(\frac{N}{2^{\ell_0-1}}\Big)^3.$$

Суммирование геометрической прогрессии даёт

$$R_1 = N^3\Big(\frac{2^6}{2^2\cdot 63}\Big(1 - \frac{1}{2^{3\ell_0}}\Big)\Big).$$

Отсюда

$$R_1 < \frac{16}{63}N^3. \tag{40}$$

С учётом времени максимально параллельного умножения двух матриц временная сложность первого шага рассматриваемой схемы сдваивания представима в виде

$$T(R_1) = t_y + \Big(\log_2\frac{N}{2}\Big)t_c = O\Big(\log_2\frac{N}{2}\Big),$$

где R_1 из (40). Это число процессоров является наибольшим для данной схемы, поскольку размерность клеток и число перемножаемых матриц уменьшается с ростом номера шага.

На каждом шаге сдваивания наибольшее число сложений находится в первой справа налево паре матриц в соответствии с их наибольшим размером. На шаге с номером $1+p$ первая пара матриц примет вид

$$\begin{bmatrix} K_{2^p} & 0 \\ 0 & 1 \end{bmatrix}[K_0], \quad p = 0, 1, \ldots, \log_2(\ell_0) - 1, \tag{41}$$

где клетка K_{2^p} имеет размерность $(N/2^{2^p}) \times (N/2^{2^p})$, клетка K_0 — $N \times N$. В максимально параллельной форме умножение матриц (41) включает один последовательный шаг парных умножений и $\log_2(N/2^{2^p})$ последовательных шагов сложения по схеме сдваивания. Отсюда суммарное число последовательных шагов максимально параллельного выполнения схемы сдваивания матриц из (37) включает $\log(\ell_0)$ бинарных умножений и число бинарных сложений, не большее

$$\log_2\frac{N}{2^{2^0}} + \log_2\frac{N}{2^{2^1}} + \log_2\frac{N}{2^{2^2}} + \ldots + \log_2\frac{N}{2^{2^{\log_2\ell_0-1}}},$$

что в сумме равно

$$\log_2\frac{N^{\log_2\ell_0}}{2^{2^{\log_2\ell_0}-1}} = \log_2\ell_0\log_2 N - \ell_0 + 1.$$

Поскольку $\ell_0 \le \log_2 N$, то число последовательных сложений не больше

$$\log_2\log_2 N \times \log_2 N + 1 \cong \log_2\log_2 N \times \log_2 N.$$

Из изложенного вытекает

Лемма 7. Матрицу преобразования вейвлета Добеши размера $N \times N$, заданного фильтрами $\{h_n\}$ и $\{g_n\}$, $n = 1, 2, \ldots, L$, $L = \text{const}$, $L \leq N$, можно вычислить параллельно за время

$$T(R_1) = O(\log_2 \log_2 N \times \log_2 N), \tag{42}$$

где число процессоров R_1 ограничено из (40).

После вычисления матрицы преобразования по схеме сдваивания за время $O(\log_2 N)$ находятся вейвлет-коэффициенты левой части (38).

Отсюда с учётом $O(\log_2 \log_2 N \times \log_2 N) + O(\log_2 N) = O(\log_2 \log_2 N \times \log_2 N)$ имеет место

Теорема 4. В условиях леммы 7 вейвлет-преобразование (38) для вейвлетов Добеши можно выполнить параллельно с временной сложностью (42), где R_1 ограничено согласно (40).

Замечание 4. С помощью схемы из леммы 7 можно вычислить матрицу Хаара размера $N \times N$ с оценкой (42). Масштабирующий фильтр для матрицы Хаара состоит из двух ненулевых элементов $h_1 = 1/\sqrt{2}$, $h_2 = 1/\sqrt{2}$, фильтр вейвлетов — из элементов $g_1 = 1/\sqrt{2}$, $g_2 = -1/\sqrt{2}$ [15].

Параллельное вычисление базиса дискретного преобразования Фурье. Схема (14)–(16) применима для параллельного нахождения базиса преобразований Фурье с оценкой $O(1)$ [5, 16]. Для этого достаточно элементы базиса вида $\sin(2\pi nk/N)$, $\cos(2\pi nk/N)$ вычислить синхронно и взаимно независимо для всех значений аргумента в каждой компоненте разложения. Например, в прямом дискретном преобразовании Фурье (ДПФ)

$$\mathrm{Re}(X(k)) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\cos(2\pi nk/N), \quad \mathrm{Im}(X(k)) = -\frac{1}{N}\sum_{n=0}^{N-1} x(n)\sin(2\pi nk/N),$$

$$n = 0, 1, \ldots, N-1, \quad k = 0, 1, \ldots, N-1,$$

число элементов каждого вида составит N для каждого k, по построению кусочно-полиномиальной схемы элемент вычисляется за время $O(1)$. Отсюда вся совокупность элементов базиса с учётом числа отсчётов и числа базисных функций определяется параллельно с этой же оценкой при наличии $O(N^2)$ процессоров. С использованием схемы сдваивания последующее ДПФ выполняется с оценкой $O(\log_2 N)$.

Таким образом, ДПФ выполняется с логарифмической временной сложностью. Как нетрудно видеть, значение N при этом может быть переменным, следовательно, оценка сохраняется в условиях динамически меняющейся размерности [9].

Детали и разновидности кусочно-полиномиальной аппроксимации базиса для преобразований Фурье излагаются в [5, 9, 16]; в [5] показана возможность распространения подхода на многомерные преобразования; в [16] приведены структура данных, программная реализация ДПФ на основе кусочно-полиномиального вычисления базиса и численный эксперимент.

Заключение. Параллельные схемы одномерных ортогональных преобразований, в числе которых пилообразное преобразование, преобразования Уолша, Хаара, Фурье и быстрое вейвлет-преобразование, строятся на основе естественного параллелизма, отличаясь при этом верхними оценками временной сложности. В частности, функции Уолша формируются с оценкой $O(\log_2 \log_2 N)$ без ограничений [17] аппаратной реализации. Преобразование Хаара выполнимо за время $O(\log_2 N)$ в отличие от известного алгоритма [18] временной сложности $O(N)$. Параллельная разновидность быстрого вейвлет-преобразования с

временем выполнения $O(\log_2 \log_2 N \times \log_2 N)$ отличается от схемы на основе блока фильтров с оценкой $O(N)$ [19]. Схема кусочно-полиномиальной аппроксимации функций применима для параллельного вычисления базиса преобразований Фурье с оценкой $O(1)$ и последующего выполнения ДПФ с оценкой $O(\log_2 N)$ [5, 9]. Ускорение в предложенных схемах достигается преимущественно за счёт избыточных затрат памяти компьютера, исключение составляет разновидность схемы Стоуна применительно к вычислению функций Радемахера.

СПИСОК ЛИТЕРАТУРЫ

1. **Миклошко Й.** Связь между алгоритмами, программами и структурой параллельных ЭВМ // Алгоритмы, математическое обеспечение и архитектура многопроцессорных вычислительных систем /Под ред. А. П. Ершова. М.: Наука, 1982. С. 6–36.
2. **Солодовников В. И.** Верхние оценки сложности решения систем линейных уравнений // Теория сложности вычислений. Ч. I. Записки научных семинаров ЛОМИ АН СССР. Л.: Наука, 1982. Т. 118. С. 159–187.
3. **Ахмед Н., Рао К. Р.** Ортогональные преобразования при обработке цифровых сигналов: Пер. с англ. /Под ред. И. Б. Фоменко. М.: Связь, 1980. 248 с.
4. **Прэтт У.** Цифровая обработка изображений. Кн. 1: Пер. с англ. /Под ред. Д. С. Лебедева. М.: Мир, 1982. 311 с.
5. **Аксайская Л. Н.** Разработка и исследование параллельных схем цифровой обработки сигналов на основе минимизации временной сложности вычисления функций: Автореф. дис. ... канд. техн. наук. Таганрог: ТТИ ЮФУ, 2008. 18 с.
6. **Ромм Я. Е., Забеглов В. В.** Параллельные алгоритмы пилообразного преобразования для цифровой обработки изображений /ТГПИ. Таганрог, 2008. 19 с. Деп. в ВИНИТИ 30.09.2008, № 783-В2008.
7. **Никитин Г. И.** Применение функций Уолша в сотовых системах связи с кодовым разделением каналов. С.-Пб.: СПбГУАП, 2003. 86 с.
8. **Ромм Я. Е.** Локализация и устойчивое вычисление нулей многочлена на основе сортировки. Ч. II // Кибернетика и системный анализ. 2007. № 2. С. 161–174.
9. **Ромм Я. Е., Фирсова С. А.** Параллельная схема дискретного и быстрого преобразований Фурье на основе полиномиального представления базиса // Изв. РАН. Сер. Математическое моделирование. 2006. **18**, № 11. С. 3–12.
10. **Stone H. S., Kogge P. M.** A parallel algorithm for the efficient solution of a general class of recurrence equations // IEEE Trans. Comput. 1973. **C22**, N 8. P. 786–792.
11. **Люстерник Л. А., Червоненкис О. А., Янпольский А. Р.** Математический анализ. Вычисление элементарных функций. М.: Физматгиз, 1963. 248 с.
12. **Ромм Я. Е.** Бесконфликтные и устойчивые методы детерминированной параллельной обработки: Дис. ... д-ра техн. наук. Таганрог, 1998. 546 с.
13. **Ромм Я. Е., Забеглов В. В.** Параллельные схемы преобразований Уолша и Хаара /ТГПИ. Таганрог, 2008. 29 с. Деп. в ВИНИТИ 30.09.2008, № 783-В2008.
14. **Mallat S.** Multiresolution approximation and wavelet orthonormal basis of $L^2(R)$ // Trans. AMS. 1989. **315**, N 1. P. 69–87.
15. **Воробьев В. И., Грибунин В. Г.** Теория и практика вейвлет-преобразования. С.-Пб.: ВУС, 1999. С. 1–204.

16. **Ромм Я. Е., Забеглов В. В.** Моделирование дискретного преобразования Фурье на основе кусочно-полиномиального вычисления базиса /ТГПИ. Таганрог, 2010. 21 с. Деп. в ВИНИТИ 28.01.2010, № 60-В2010.

17. **Визор Я. Е., Чичирин Е. Н., Семотюк М. В.** Формирование функций Уолша для многоканальных систем реального времени // Проблеми інформатизації та управління. 2009. **4**, № 28. С. 24–27.

18. **Chan K. P., Fu A. W.** Effcient time series matching by wavelets // Proc. of the 15th Intern. Conf. on Data Engineering. 1999. P. 126–133.

19. **Столниц Э., ДеРоуз Т., Салезин Д.** Вейвлеты в компьютерной графике: Пер. с англ. Ижевск: НИЦ «Регулярная и хаотическая динамика», 2002. 272 с.

Поступила в редакцию 19 мая 2010 г.