Calculating the *n*-Dimensional Fast Fourier Transform

V. S. Tutatchikov, O. I. Kiselev, and M. V. Noskov

Institute of Space and Information Technology, Siberian Federal University, ul. Akademika Kirenskogo 26, Krasnoyarsk, 660074 Russia

e-mail: mvnoskov@yandex.ru, vtutatchikov@mail.ru

Abstract—The one-dimensional fast Fourier transform (FFT) is the most popular tool for calculating the multidimensional Fourier transform. As a rule, to estimate the *n*-dimensional FFT, a standard method of combining one-dimensional FFTs, the so-called "by rows and columns" algorithm, is used in the literature. For fast calculations, different researchers try to use parallel calculation tools, the most successful of which are searches for the algorithms related to the computing device architecture: cluster, video card, GPU, etc. [1, 2]. The possibility of paralleling another algorithm for FFT calculation, which is an *n*-dimensional analog of the Cooley-Tukey algorithm [3, 4], is studied in this paper. The focus is on studying the analog of the Cooley-Tukey algorithm because the number of operations applied to calculate the *n*-dimensional FFT is considerably less than in the conventional algorithm $nN^n\log_2 N$ of addition operations and $1/2N^{n+1}\log_2 N$ of multiplication operations of addition operations and $\frac{2^n-1}{2^n}N^n\log_2 N$ of multiplication operations against: $N^{n+1}\log_2 N$ of addition operations and $1/2N^{n+1}\log_2 N$ of in combining one-dimensional FFTs.

Keywords: multiple fast Fourier transform, Cooley-Tukey FFT, parallel FFT algorithm.

DOI: 10.1134/S1054661813030140

RECURRENT SCHEME OF THE *n*-DIMENSIONAL ANALOG OF COOLEY-TUKEY ALGORITHM

The recurrent scheme of the *n*-dimensional analogue of Cooley-Tukey algorithm can be introduced as follows.

Let a periodic (the period is $N = 2^s$, $s \in Z$) complex-valued function of the integral argument be termed the *n*-dimensional periodic signal $x(j_1, ..., j_n)$ at fixed N.

A set of signals C_N^n, combined with the operations of addition of two signals x_1, x_2

$$y(j) = x_1(j) + x_2(j) \tag{1}$$

and multiplication of signal x by the complex number c

$$y(j) = cx(j), \tag{2}$$

where $x(j)$ is the x signal counting in the point $j \in \mathbb{Z}^n$, becomes a linear complex space. The null element in C_N^n is a signal O such that $O(j) = 0$ for all $j \in Z^n$. Let us introduce the scalar product and norm in C_N^n:

$$\langle x, y\rangle = \sum_{j \in B_n(N)} x(j)\overline{y(j)},$$
$$\|x\| = \langle x, x\rangle^{1/2}, \tag{3}$$

where $B_n(N)$ is a set of integral vectors from $[0, N-1]^n$.

Let us denote $w_N = e^{\frac{2\pi i}{N}}$. By an *n*-multiple (*n*-dimensional) discrete Fourier transform (DFT) is meant the reflection of F_N: $C_N^n \longrightarrow C_N^n$ matching of a signal X containing the values

$$X(j) = \sum_{t \in B_n(N)} x(t)w_n^{-(j,t)}, \tag{4}$$

where $j \in B_n(N)$, with the signal x.

Let $N = 2^s$, $N_\nu = 2^{s-\nu}$, $\Delta_\nu = 2^{\nu-1}$. We construct a recurrent sequence of bases $f_0, f_1, ..., f_s$, where f_t is the tth basis consisting of N^n bases $f_t(k)$, $k \in B_n(N)$. The value of signal $f_t(k)$ in the sampling $j = (j_1, ..., j_n)$, $j \in B_n(N)$ will be denoted as $f_t(k, j)$.

We designate the set of integral vectors from $[0, N_\nu - 1]^n$ as $B_n^1(N)$ and the set of integral vectors from $[0, \Delta_\nu - 1]^n$ as $B_n^2(N)$.

Let j, being an integer number from the set $J = \{0, 1, ..., 2^\nu - 1\}$, be presented in the binary system in the form $j_{\nu-1}2^{\nu-1} + ... + j_1 2 + j_0$, where $j_i = 0, 1$ for all $i = 0, ..., \nu - 1$. The vector $(j_{\nu-1}, ..., j_1, j_0)_2$ will be said to be the binary code of the number j. Let us correlate the number $j_1 \in J$, which is specified by the binary

Received March 2, 2012

code $(j_0, j_1, ..., j_{v-1})_2$, with the number j. The repositioning $\mathrm{rev}_v j = j_1$ of the set J is called reversed.

The recurrent scheme for calculating the signal spectrum $x \in \mathbb{C}_N^n$ is

$$\begin{aligned} x_0(k) &= x(\mathrm{rev}_s k_1, \ldots, \mathrm{rev}_s k_n), \\ x_v(l_1 + \sigma_1\Delta_v + p_1\Delta_{v+1}, \ldots, l_n &+ \sigma_n\Delta_v, \ldots, 2\Delta_v p_n\Delta_{v+1}) \\ = \sum_{\tau_1}^{1} \cdots \sum_{\tau_n}^{1} w_{\Delta_{v+1}}^{\sum_{i=0}^{n} \tau_i(l_i+\sigma_i\Delta_v)} & x_{v-1}(l_1 + 2\Delta_v p_1 \\ + \tau_1\Delta_v, \ldots, l_n &+ 2\Delta_v p_n + \tau_n\Delta_v), \end{aligned} \quad (5)$$

where $p = (p_1, ..., p_n)$, $p \in B_n^1(N)$, $l = (l_1, ..., l_2)$, $l \in B_n^2(N)$, $v = 1, ..., s$ and $\sigma_1, ..., \sigma_n$ are 0 and 1.

PARALLEL ALGORITHM OF n-DIMENSIONAL FFT BY THE ANALOG OF THE COOLEY-TUKEY ALGORITHM

As a preliminary, let us consider the parallel algorithm of two-dimensional FFT, which is the analog to the Cooley-Tukey algorithm. We apply this algorithm to calculate recurrent scheme (5) at each step:

(1) The initial signal x_{v-1}, $v = 1, ..., s$ is represented in the following form:

$$\begin{aligned} x_{v-1}(k_1, k_2) &= x'_{v-1}(2k'_1, 2k'_2) + x'_{v-1}(2k'_1 + 1, 2k'_2) \\ &+ x'_{v-1}(2k'_1, 2k'_2 + 1) + x'_{v-1}(2k'_1 + 1, 2k'_2 + 1), \end{aligned} \quad (6)$$

where the subsignal x'_{v-1} contains the components of signal x_{v-1} with different evennesses of coordinates $k_1, k_2, k'_1, k'_2 = 0, ..., N/2$;

(2) We choose all minors of the second order in the following form:

$$\begin{pmatrix} a_{ij} & a_{ij+2^v} \\ a_{i+2^v} & a_{i+2^v j+2^v} \end{pmatrix}, \quad (7)$$

where $a_{ij} \in x'_{v-1}$, while the elements a_{ij} are not repeated for different minors.

(3) The major process sends different pairs of rows i and $i + 2^v$ containing minors (7) to each computational process.

(4) Computational processes calculate the FFT above minors (7) contained in the obtained rows, record the result in these rows for the place of calculated elements, and send it to the major process.

(5) The major process accepts the rows and records them into the initial positions of the matrix.

After calculating all three steps, the matrix is normalized by the major process.

During transition to the three-dimensional case, certain points of the algorithm change. At the first one, the initial signal is decomposed into eight subsignals with an evenness of the coordinate differing from four:

$$\begin{aligned} x_{v-1}(k_1, k_2, k_3) &= x'_{v-1}(2k'_1, 2k'_2, 2k'_3) \\ + x'_{v-1}(2k'_1 + 1, 2k'_2, 2k'_3) &+ x'_{v-1}(2k'_1, 2k'_2 + 1, 2k'_3) \\ + x'_{v-1}(2k'_1 + 1, 2k'_2 + 1, 2k'_3) & \\ + x'_{v-1}(2k'_1, 2k'_2, 2k'_3 + 1) & \\ + x'_{v-1}(2k'_1 + 1, 2k'_2, 2k'_3 + 1) & \\ + x'_{v-1}(2k'_1, 2k'_2 + 1, 2k'_3 + 1) & \\ + x'_{v-1}(2k'_1 + 1, 2k'_2 + 1, 2k'_3 + 1), & \end{aligned} \quad (8)$$

where $k'_1, k'_2, k'_3 = 0, ..., N/2$.

At the second point, we select three-dimensional second-order matrices:

$$\begin{pmatrix} b_k \\ b_{k+2^{s-1}} \end{pmatrix}, \quad b_k = \begin{pmatrix} a_{ijk} & a_{ij+2^{s-1}k} \\ a_{i+2^{s-1}jk} & a_{i+2^{s-1}j+2^{s-1}k} \end{pmatrix}, \quad b_{k+2^{s-1}} = \begin{pmatrix} a_{ijk+2^{s-1}} & a_{ij+2^{s-1}k+2^{s-1}} \\ a_{i+2^{s-1}jk+2^{s-1}} & a_{i+2^{s-1}j+2^{s-1}k+2^{s-1}} \end{pmatrix}. \quad (9)$$

At the third and subsequent points, the major and calculation processes will operate the planes b_k and $b_{k+2^{s-1}}$ rather than rows.

The transition to the n-dimensional case can be implemented in a similar manner. Therefore, at the first point, the initial signal is decomposed into 2^n subsignals with different evenness. At the second point, we choose the n-dimensional second-order matrices. At the third and subsequent points, the processes works with hyperplanes.

Let us consider the two-dimensional FFT more fully.

TWO-DIMENSIONAL FFT

The two-dimensional FFT is used for analyzing two-dimensional signals and image processing (increasing definition quality and brightness).

Let us consider a signal f that is a two-dimensional periodic signal with a period of 2^s over two coordinates. Samplings are specified as $f(x, y)$, where $x, y = 0 : 2^s$. The discrete Fourier transform for the signal f is given by the following formula:

$$F(a, b) = \sum_{x=0}^{2^s-1}\sum_{y=0}^{2^s-1} f(x, y)e^{\frac{2\pi i(ax+by)}{2^s}}. \quad (10)$$

The two-dimensional FFT F can be calculated by using one-dimensional FFTs:

$$F(a, b) = \sum_{x=0}^{2^s-1}\left[\sum_{y=0}^{2^s-1} f(x, y)e^{\frac{2\pi iax}{2^s}}\right]e^{\frac{2\pi iby}{2^s}}. \quad (11)$$

Sums in square brackets are one-dimensional calculations of the FFT over two coordinates of signal f. Let us transform the given formula with decomposing coordinates into even and odd components:

$$F(a, b) = \sum_{x=0}^{2^s-1}\sum_{y=0}^{2^s-1} f(x, y, z)e^{\frac{2\pi iax}{2^s}}e^{\frac{2\pi iby}{2^s}}$$

$$= \sum_{x_1=0}^{2^{s-1}-1}\sum_{y_1=0}^{2^{s-1}-1}\sum_{z_1=0}^{2^{s-1}-1} f(2x_1, 2y_1, 2z_1)e^{\frac{2\pi iax_1}{2^{s-1}}}e^{\frac{2\pi iby_1}{2^{s-1}}}e^{\frac{2\pi icz_1}{2^{s-1}}}$$

$$+ e^{\frac{\pi ia}{2^s}}\sum_{x_1=0}^{2^{s-1}}\sum_{y_1=0}^{2^{s-1}-1} f(2x_1+1, 2y_1)e^{\frac{2\pi iax_1}{2^{s-1}}}e^{\frac{2\pi iby_1}{2^{s-1}}}$$

$$+ e^{\frac{\pi ib}{2^s}}\sum_{x_1=0}^{2^{s-1}-1}\sum_{y_1=0}^{2^{s-1}-1} f(2x_1, 2y_1+1)e^{\frac{2\pi iax_1}{2^{s-1}}}e^{\frac{2\pi iby_1}{2^{s-1}}} \quad (12)$$

$$+ e^{\frac{\pi ia}{2^s}}e^{\frac{1\pi ib}{2^s}}\sum_{x_1=0}^{2^{s-1}-1}\sum_{y_1=0}^{2^{s-1}-1} f(2x_1+1, 2y_1+1)e^{\frac{2\pi iax_1}{2^{s-1}}}e^{\frac{2\pi iby_1}{2^{s-1}}}$$

$$= F_{0,0}(a, b) + e^{\frac{\pi ia}{2^s}}F_{1,0}(a, b)$$

$$+ e^{\frac{\pi ib}{2^s}}F_{0,1}(a, b) + e^{\frac{\pi ia}{2^s}}e^{\frac{\pi ib}{2^s}}F_{1,1}(a, b),$$

where $a, b = 0 : 2^s - 1$.

It can be shown that $e^{\frac{\pi ia}{2^s}}$ is symmetric with respect to $a = 2^{s-1}$:

$$e^{\frac{\pi i(s^{s-1}+t)}{2^s}} = e^{\frac{\pi i2^{s-1}}{2^s}}e^{\frac{\pi it}{2^s}} = -e^{\frac{\pi it}{2^s}}, \quad (13)$$

where $t = 0 : 2^{s-1} - 1$. By analogy, $e^{\frac{\pi ib}{2^s}}$ is symmetric with respect to $b = 2^{s-1}$. Then, from (3) and (4) we obtain

$$F(a, b) = F_{0,0}(a, b) + e^{\frac{\pi ia}{2^s}}F_{1,0}(a, b)$$

$$+ e^{\frac{\pi ib}{2^s}}F_{0,1}(a, b) + e^{\frac{\pi ia}{2^s}}e^{\frac{\pi ib}{2^s}}F_{1,1}(a, b),$$

$$F(a + 2^{s-1}, b) = F_{0,0}(a + 2^{s-1}, b)$$

$$- e^{\frac{\pi ia}{2^s}}F_{1,0}(a + 2^{s-1}, b) + e^{\frac{\pi ib}{2^s}}F_{0,1}(a + 2^{s-1}, b)$$

$$- e^{\frac{\pi ia}{2^s}}e^{\frac{\pi ib}{2^s}}F_{1,1}(a + 2^{s-1}, b + 2^{s-1}),$$

$$F(a, b + 2^{s-1}) = F_{0,0}(a, b + 2^{s-1})$$

$$+ e^{\frac{\pi ia}{2^s}}F_{1,0}(a, b + 2^{s-1}) - e^{\frac{\pi ib}{2^s}}F_{0,1}(a, b + 2^{s-1}) \quad (14)$$

$$- e^{\frac{\pi ia}{2^s}}e^{\frac{\pi ib}{2^s}}F_{1,1}(a, b + 2^{s-1}),$$

$$F(a + 2^{s-1}, b + 2^{s-1})$$

$$= F_{0,0}(a + 2^{s-1}, b + 2^{s-1})$$

$$- e^{\frac{\pi ia}{2^s}}F_{1,0}(a + 2^{s-1}, b + 2^{s-1})$$

$$- e^{\frac{\pi ib}{2^s}}F_{0,1}(a + 2^{s-1}, b + 2^{s-1})$$

$$+ e^{\frac{\pi ia}{2^s}}e^{\frac{\pi ib}{2^s}}F_{1,1}(a + 2^{s-1}, b + 2^{s-1}),$$

where $a, b = 0 : 2^{s-1} - 1$.

A number of complex multiplications and additions of the two-dimensional FFT algorithm, which is an analog of the Cooley-Tukey one, are $3/4N^2\log_2 N$ and $2N^2\log_2 N$ in comparison with the calculation method using the one-dimensional FFT, where $1/2N^3\log_2 N$ of multiplications and $N^3\log_2 N$ of additions.

RESULTS

To test the algorithm, a program was written in the C++ language using the MPI library. The program realizes two algorithms: the two-dimensional FFT by rows and columns (2) and the two-dimensional FFT by the analog of the Cooley-Tukey algorithm (7). The two-dimensional FFT by rows and columns is used in such mathematical packages as MatLab and MathCad, as well as the open library fftw. Only the algorithm by rows and columns (fftw), the simplest in terms of implementation, acts as an algorithm parallel to FFT. Testing was conducted on the supercomputer of the Institute of Space and Information Technology of Siberian Federal University, which includes 224 IBM Blade HS21computational nodes. Each node contains 16 Gb of RAM and two quad-core CPU Xeon E5345@2.33 GHz processors. During testing,

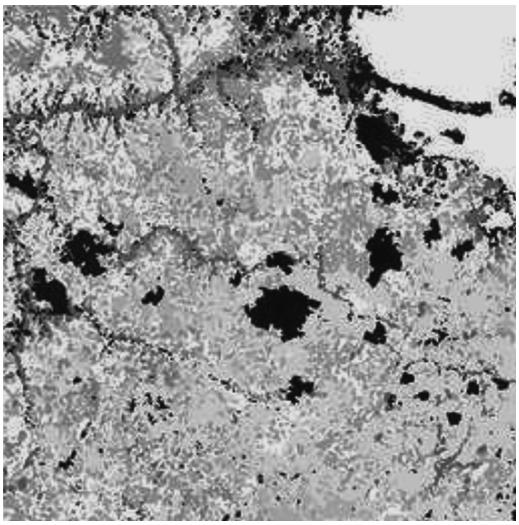

Fig. 1. Example of initial signal.

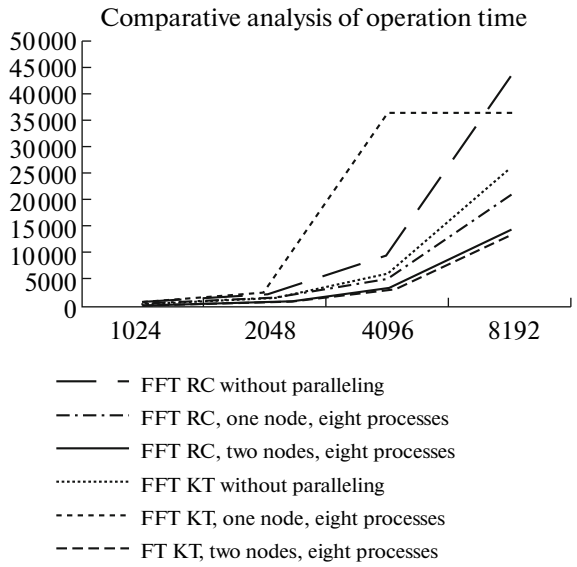

Fig. 2. Comparison of operation times of different algorithms.

the program operation time was measured on one node without paralleling, on one node with paralleling for eight processes, and on two nodes with paralleling for eight processes.

Table 1. Result of testing one node

N	P	FFT RC	FFT KT
1024	1	490	310
	2	310	310
	4	220	270
	8	180	270
	16	180	350
2048	1	2300	1330
	2	1550	1260
	4	1060	950
	8	840	850
	16	850	1000
4096	1	9880	5850
	2	6240	4590
	4	4370	3590
	8	3430	3060
	16	3740	3390
8192	1	43210	25190
	2	26990	19160
	4	18550	14870
	8	14520	13130
	16	14090	11700

Note: $N = 2^s$, $N \times N$ is the number of signal samplings $x(j_1, j_2)$; P is the number of started MPI-processes; FFT RC is the FFT by rows and column; FFT KT is the FFT by Kuli–T'uki analog.

Table 1 shows the results of the average measuring of time in milliseconds at the start on the first node with paralleling for eight processes. Space shots were used as the initial signal; an example is shown in Fig. 1.

The calculation result is given in milliseconds.

THREE-DIMENSIONAL FFT

The three-dimensional FFT is executed by a scheme similar to the two-dimensional case:

$$\begin{aligned} F(a,b,c) &= g(2x_1, 2y_1, 2z_1) \\ &+ e^{\frac{2\pi i a}{2^s}} g(2x_1+1, 2y_1, 2z_1) \\ &+ e^{\frac{2\pi i b}{2^s}} g(2x_1, 2y_1+1, 2z_1) \\ &+ e^{\frac{2\pi i c}{2^s}} g(2x_1, 2y_1, 2z_1+1) \\ &+ e^{\frac{2\pi i (a+b)}{2^s}} g(2x_1+1, 2y_1+1, 2z_1) \\ &+ e^{\frac{2\pi i (b+c)}{2^s}} g(2x_1, 2y_1+1, 2z_1+1) \\ &+ e^{\frac{2\pi i (a+c)}{2^s}} g(2x_1+1, 2y_1, 2z_1+1) \\ &+ e^{\frac{2\pi i (a+b+c)}{2^s}} g(2x_1+1, 2y_1+1, 2z_1+1), \end{aligned} \quad (15)$$

$$g(x,y,z) = \sum_{x=0}^{2^s}\sum_{y=0}^{2^s}\sum_{z=0}^{2^s} f(x,y,z) e^{\frac{2\pi i (ax_2 + by_2 + cz)}{2^s}},$$

Table 2. Comparison of operation times of algorithms of three-dimensional FFT using combination of one-dimensional and three-dimensional FFT by Cooley–Tukey analog, in seconds

Size	FFT with using the combination of 1D FFT	FFT by Cooley–Tukey analog
$32 \times 32 \times 32$	0.006	0.002
$64 \times 64 \times 64$	0.045	0.019
$128 \times 128 \times 128$	0.298	0.134
$256 \times 256 \times 256$	2.850	1.225
$512 \times 512 \times 512$	29.840	10.658

where $g(x, y, z)$ is the three-dimensional FFT of a signal of even and odd components of three coordinates of thinned signal; $x_2 = x\text{div}2$, $y_2 = y\text{div}2$, $z_2 = z\text{div}2$, x_2, y_2, z_2 is the result of integer division by 2.

The number of complex multiplications and additions of the two-dimensional FFT algorithm, similar to the Cooley-Tukey one, are $7/8N^3\log_2 N$ and $3N^3\log_2 N$ in comparison with the calculation method using the one-dimensional FFT, where $N^3\log_2 N$ of multiplications and $N^3\log_2 N$ of additions.

To test the algorithm, a program in C++ was written. It realizes two algorithms: the three-dimensional FFT using combinations of one-dimensional FFTs and FFT by the Kuli–T'uki analog. Testing was conducted on a computer with a CPU Intel Core i5 2400 GHz processor, 4 Gb of RAM, and Windows 7 OS. The operation speed of the two algorithms was in seconds. The results are given in Table 2.

As a result of research work, the algorithm of three-dimensional FFT by the Cooley-Tukey analog was attained. It works significantly faster than the algorithm of calculating the three-dimensional FFT using combination of one-dimensional FFTs.

REFERENCES

1. T. Jansen, B. Rymon-Lipinski, N. Hanssen, and E. Keeve, "Fourier volume rendering on the GPU using a split-stream-FTT," in *Proc. Vision, Modeling, and Visualization Conf.* (Stanford, 2004).
2. S. Pissis, "Parallel Fourier transformations using shared memory nodes," MSc in High Performance Computing (Univ. of Edinburgh, 2008).
3. D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing* (Prentice Hall, 1983).
4. A. V. Starovoitov, "On multidimensional analog of Tukey-Cooley algorithm," Vestn. Siberian State Aerospace Univ., No. 1 (27), 69–73 (2010).

Translated by A. Evseeva

Valera S. Tutatchikov. Born 1988. Graduated from Institute of Space and Information Technology, Siberian Federal University in 2011. Undergraduate at Institute of Space and Information Technology, Siberian Federal University. Scientific interests: fast Fourier transform, parallel algorithms. Author of ten papers.

Mikhail V. Noskov. Born 1947. Graduated from Krasnoyarsk State Pedagogical Institute in 1969. Received candidate's degree in 1984 and doctorar degree in 1992. Scientific interests: cubature formulas, mathematics teaching method. Author of 53 papers.

Oleg I. Kiselev. Born 1965. Graduated from Krasnoyarsk State Polytechnical Institute, Siberian Federal University in 1987. Scientific interests: signal processing. Author of six papers.