Bartosz ANDREATTO, Aleksandr CARIOW

West Pomeranian University of Technology, Szczecin

A fast algorithm for multiresolution discrete Fourier transform

***Abstract**. The paper presents a fast algorithm for the calculation of a multiresolution discrete Fourier transform. The presented approach is based on the realization of the Fast Fourier Transform for each frequency resolution level. This algorithm allows reducing the number of complex multiplications and additions compared to the method consisting in the multiplication between the input signal expressed as a column vector and the matrix of discrete exponential functions.*

***Streszczenie.** W artykule przedstawiono szybki algorytm wyznaczania wielorozdzielczej dyskretnej transformaty Fouriera. Zaprezentowane podejście opiera się na realizacji algorytmu szybkiej transformacji Fouriera na każdym z analizowanych poziomów rozdzielczości częstotliwościowej. Algorytm ten cechuje się zredukowaną liczbą operacji mnożenia oraz dodawania liczb zespolonych, w porównaniu do algorytmu opierającego się na mnożeniu wektora sygnału wejściowego przez macierz dyskretnych funkcji wykładniczych. (**Szybki algorytm wielorozdzielczej dyskretnej transformaty Fouriera**).*

Słowa kluczowe: szybki algorytm, wielorozdzielcza dyskretna transformacja Fouriera, szybka transformacja Fouriera, notacja macierzowa.
Keywords: fast algorithm, multiresolution discrete Fourier transforms, Fast Fourier Transform, matrix notation

Introduction

Transform domain approach for digital signal processing is useful to bring out the hidden information, which may not be explicitly available when the signal is represented in original domain. In recent years, discrete orthogonal transforms have attracted a considerable amount of attention, resulting in many applications of signal processing [1-3]. However, conventional transforms focus on the “frequency” domain, can’t analyze the variations of frequency with time. Nevertheless, in many applications, there is the task of analyzing the frequency components of signals with reference to time. In addition, it is often necessary analysis of selected segments of the signal with different resolutions. Traditional discrete orthogonal transforms these properties do not possess. All the above advantages has a multiresolution discrete Fourier transform (MR DFT) [4]. It is necessary to mark that the calculation of MR DFT requires implementation of plenty of arithmetic operations. In [5] described an algorithm that reduces the number of arithmetic operations in calculating the MR DFT. But in this publication is not taken into account all aspects, which can be used in the implementation of the considered method. In particular, it is not fully explored the possibility of rationalization of the computational process when implementing the studied method. The authors of paper [5] did not show any graph illustrating the organization of the computational process at least for the simplest example. It is therefore often difficult to understand the solutions proposed by the authors of the work. In this regard, the proposed recommendations are not suitable for direct use. Therefore, in this paper we presented a clear and explicit algorithm for computing the MR DFT. The algorithm is also well suited for parallel processing because of the modular nature of time-space structure of computing process and regularity of its structural fragments. In describing the algorithm used matrix notation that enables simple and clear to provide features of computational process.

The vectorized form of the multiresolution discrete Fourier transform may be defined as:

$$\mathbf{Y}_{mN\times 1} = \bigoplus_{i=1}^{m}(\mathbf{I}_{2^{m-i}} \otimes \mathbf{E}_{2^i})\mathbf{P}_{mN\times N}\mathbf{X}_{N\times 1}, \tag{1}$$

where: $\mathbf{X}_{N\times 1} = [x_0, x_1, ..., x_{N-1}]^{\mathrm{T}}$ – is N-dimensional input vector, $\mathbf{Y}_{mN\times 1} = [\mathbf{Y}^{(1)}_{N\times 1}, \mathbf{Y}^{(2)}_{N\times 1}, ..., \mathbf{Y}^{(m)}_{N\times 1}]^{\mathrm{T}}$ – mN-dimensional output vector, whose elements are vectors $\mathbf{Y}^{(i)}_N = [y^{(i)}_0, y^{(i)}_1, ..., y^{(i)}_{N-1}]^{\mathrm{T}}$ corresponding to all spectral components of signal at different resolution levels, $i = 1,2,...,m$ – indicate the expected resolution level, and $N = 2^m$. $\mathbf{E}_{2^i}$ – is the matrix of discrete exponential functions [6], whose elements are $w^{n,k}_{2^i} = \exp(-j2\pi nk/2^i)$, where j is the imaginary unit, satisfying $j^2 = -1$. $\mathbf{I}_N$ – is the identity $N\times N$ matrix and signs “$\otimes$” and “$\oplus$” – denote a tensor product and direct sum of two matrices respectively [7]. The matrix of the extension of input vector is defined as:

$$\mathbf{P}_{mN\times N} = \mathbf{1}_{m\times 1} \otimes \mathbf{I}_N, \tag{2}$$

where $\mathbf{1}_{m\times 1}$ – is the $m\times 1$ matrix whose elements are units.

As can be seen, a schoolbook way of computing the multiresolution discrete Fourier transform in accordance with equation (1) requires $\alpha_\times$ multiplications and α_+ additions in complex numbers. These values are defined in a following way:

$$\alpha_\times = \sum_{i=1}^{m}(2^{m-i}\cdot(2^i)^2) = 2N^2 - 2N,$$

$$\alpha_+ = \sum_{i=1}^{m}(2^{m-i}\cdot 2^i \cdot (2^i - 1)) = 2N^2 - (m+2)N.$$

In the rest of this paper we describe in detail the fast algorithm for the multiresolution discrete Fourier Transform with reduced number of arithmetic operations compared to the naive method.

Development of the algorithm

The main idea of the proposed algorithm is a realization of the Fast Fourier Transform for selected segments of the signal at the current resolution level. At this point, we introduce some special matrices, used in the synthesis of a computational procedure.

The matrix $\mathbf{A}^{(i)}_{mN}$, which refers to the i-th iteration of the algorithm, is defined as follows:

$$\mathbf{A}^{(i)}_{mN} = \mathbf{I}_{(i-1)N} \oplus \mathbf{S}^{(i)}_{(m-i+1)N},$$

where the matrix of an algebraic summation is defined in a following way:

$$\mathbf{S}^{(i)}_{(m-i+1)N} = \bigoplus_{j=1}^{m-i+1} (\mathbf{I}_{2^{m-j}} \otimes \mathbf{H}_2 \otimes \mathbf{I}_{2^{j-1}}) ,$$

where $\mathbf{H}_2$ denotes the 2×2 Hadamard matrix:

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} .$$

The definition of the matrix $\mathbf{G}^{(i)}_{mN}$ is presented below:

$$\mathbf{G}^{(i)}_{mN} = \mathbf{I}_{(i-1)N} \oplus \mathbf{D}^{(i)}_{(m-i+1)N} ,$$

$$\mathbf{D}^{(i)}_{(m-i+1)N} = \bigoplus_{j=1}^{m-i+1} (\mathbf{I}_{2^{m-j}} \otimes \widetilde{\mathbf{D}}^{(j)}_{2^j}) .$$

The matrix $\widetilde{\mathbf{D}}^{(j)}_{2^j}$, which determine a position and a size of “the butterflies” is defined in a following way:

$$\widetilde{\mathbf{D}}^{(j)}_{2^j} = \mathbf{I}_{2^{j-1}} \oplus diag(w^0_{2^j}, w^1_{2^j}, ..., w^{2^{j-1}-1}_{2^j}) ,$$

where w^k_n is a twiddle factor, which can be expressed as:

$$w^k_n = e^{-j\frac{2\pi k}{n}} .$$

The matrix shuffling the data takes a following form:

$$\mathbf{\Gamma}_{mN} = \bigoplus_{i=1}^{m} (\mathbf{I}_{2^{m-i}} \otimes \mathbf{T}^{(i)}_{2^i}) ,$$

where $\mathbf{T}^{(i)}_{2^i}$ is the matrix shuffling the components related to the i -th resolution level:

$$\mathbf{T}^{(i)}_{2^i} = [t_{k,n}] \text{, for } k = \overline{1,2^i} \text{ , } n = \overline{1,2^i} .$$

Nonzero elements of the bit-reversal permutation $\mathbf{T}^{(i)}_{2^i}$ are defined in a following way:

$$t_{k,\langle n-1\rangle_2+1} = 1 \text{, for } k = n = \overline{1,2^i} ,$$

where $\langle p \rangle_2$ represent a binary inversion of number p:

$$p = \sum_{i=0}^{m-1} p_i \cdot 2^i \text{ , } p_i \in \{0,1\} ,$$

for:

$$\langle p \rangle_2 = \sum_{i=0}^{m-1} p_{m-i-1} \cdot 2^i \text{ , } p_i \in \{0,1\} .$$

Taking into account all the above information, we can write the procedure for fast computation of the multiresolution discrete Fourier transform:

$$\begin{aligned} \mathbf{Y}_{mN\times 1} = \mathbf{\Gamma}_{mN} (\mathbf{D}^{(m)}_{mN}\mathbf{A}^{(m)}_{mN})(\mathbf{D}^{(m-1)}_{mN}\mathbf{A}^{(m-1)}_{mN}) \times \cdots \\ \cdots \times (\mathbf{D}^{(1)}_{mN}\mathbf{A}^{(1)}_{mN}) \mathbf{P}_{mN\times N} \mathbf{X}_{N\times 1} . \end{aligned} \tag{3}$$

Below we consider an example for computing the MR DFT for $N = 8$ ($m = 3$). The computational procedure for this example takes a following form:

$$\mathbf{Y}_{24\times 1} = \mathbf{\Gamma}_{24} (\mathbf{D}^{(3)}_{24}\mathbf{A}^{(3)}_{24})(\mathbf{D}^{(3)}_{24}\mathbf{A}^{(3)}_{24})(\mathbf{D}^{(3)}_{24}\mathbf{A}^{(3)}_{24}) \mathbf{P}_{24\times 8} \mathbf{X}_{8\times 1} ,$$

where:

$$\mathbf{X}_{8\times 1} = [x_0, x_1, ..., x_7]^{\mathrm{T}} ,$$

$$\mathbf{Y}_{24\times 1} = [\mathbf{Y}^{(1)}_{8\times 1}, \mathbf{Y}^{(2)}_{8\times 1}, \mathbf{Y}^{(3)}_{8\times 1}]^{\mathrm{T}} ,$$

$$\mathbf{Y}^{(i)}_{8\times 1} = [y^{(i)}_0, y^{(i)}_1, ..., y^{(i)}_7]^{\mathrm{T}} .$$

The matrix $\mathbf{P}_{24\times 8}$ according to formula (2) can be expressed as:

$$\mathbf{P}_{24\times 8} = \mathbf{1}_{3\times 1} \otimes \mathbf{I}_8 = \left[\begin{array}{c} \mathbf{I}_8 \\ \hdashline \mathbf{I}_8 \\ \hdashline \mathbf{I}_8 \end{array}\right] .$$

Next we present the corresponding matrices in each iteration of the algorithm.

First iteration

$$\mathbf{A}^{(1)}_{24} = \mathbf{S}^{(1)}_{24} = \hat{\mathbf{S}}^{(1)}_8 \oplus \hat{\mathbf{S}}^{(2)}_8 \oplus \hat{\mathbf{S}}^{(3)}_8 = \left[\begin{array}{c:c:c} \hat{\mathbf{S}}^{(1)}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \hat{\mathbf{S}}^{(2)}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{S}}^{(3)}_8 \end{array}\right] ,$$

$$\hat{\mathbf{S}}^{(1)}_8 = \mathbf{I}_4 \otimes \mathbf{H}_2 = \left[\begin{array}{c:c:c:c} \mathbf{H}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 \\ \hdashline \mathbf{0}_2 & \mathbf{H}_2 & \mathbf{0}_2 & \mathbf{0}_2 \\ \hdashline \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{H}_2 & \mathbf{0}_2 \\ \hdashline \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{H}_2 \end{array}\right] ,$$

$$\hat{\mathbf{S}}^{(2)}_8 = \mathbf{I}_2 \otimes \mathbf{H}_2 \otimes \mathbf{I}_2 = \left[\begin{array}{c:c} \begin{array}{c:c} \mathbf{I}_2 & \mathbf{I}_2 \\ \hdashline \mathbf{I}_2 & -\mathbf{I}_2 \end{array} & \mathbf{0}_4 \\ \hdashline \mathbf{0}_4 & \begin{array}{c:c} \mathbf{I}_2 & \mathbf{I}_2 \\ \hdashline \mathbf{I}_2 & -\mathbf{I}_2 \end{array} \end{array}\right] ,$$

$$\hat{\mathbf{S}}^{(3)}_8 = \mathbf{H}_2 \otimes \mathbf{I}_4 = \left[\begin{array}{c:c} \mathbf{I}_4 & \mathbf{I}_4 \\ \hdashline \mathbf{I}_4 & -\mathbf{I}_4 \end{array}\right] .$$

$$\mathbf{G}^{(1)}_{24} = \mathbf{D}^{(1)}_{24} = \hat{\mathbf{D}}^{(1)}_8 \oplus \hat{\mathbf{D}}^{(2)}_8 \oplus \hat{\mathbf{D}}^{(3)}_8 = \left[\begin{array}{c:c:c} \hat{\mathbf{D}}^{(1)}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \hat{\mathbf{D}}^{(2)}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{D}}^{(3)}_8 \end{array}\right] ,$$

$$\hat{\mathbf{D}}_8^{(1)} = \mathbf{I}_4 \otimes (\mathbf{I}_1 \oplus diag(w_2^0)) =$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_2^0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_2^0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_2^0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_2^0 \end{bmatrix}$$

$$\hat{\mathbf{D}}_8^{(2)} = \mathbf{I}_2 \otimes (\mathbf{I}_2 \oplus diag(w_4^0, w_4^1)) =$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_4^0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_4^1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_4^0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_4^1 \end{bmatrix},$$

$$\hat{\mathbf{D}}_8^{(3)} = \mathbf{I}_4 \oplus diag(w_8^0, w_8^1, w_8^2, w_8^3) =$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_8^0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_8^1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_8^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_8^3 \end{bmatrix}.$$

Second iteration

$$\mathbf{A}_{24}^{(2)} = \mathbf{I}_8 \oplus \mathbf{S}_{16}^{(2)} = \mathbf{I}_8 \oplus \hat{\mathbf{S}}_8^{(1)} \oplus \hat{\mathbf{S}}_8^{(2)} = \left[\begin{array}{c:c:c} \mathbf{I}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \hat{\mathbf{S}}_8^{(1)} & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{S}}_8^{(2)} \end{array}\right]$$

$$\mathbf{G}_{24}^{(2)} = \mathbf{I}_8 \oplus \mathbf{D}_{16}^{(2)} = \mathbf{I}_8 \oplus \hat{\mathbf{D}}_8^{(1)} \oplus \hat{\mathbf{D}}_8^{(2)} = \left[\begin{array}{c:c:c} \mathbf{I}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \hat{\mathbf{D}}_8^{(1)} & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{D}}_8^{(2)} \end{array}\right]$$

Third iteration

$$\mathbf{A}_{24}^{(3)} = \mathbf{I}_{16} \oplus \mathbf{S}_8^{(3)} = \mathbf{I}_{16} \oplus \hat{\mathbf{S}}_8^{(1)} = \left[\begin{array}{c:c:c} \mathbf{I}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{I}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{S}}_8^{(1)} \end{array}\right],$$

$$\mathbf{G}_{24}^{(3)} = \mathbf{I}_{16} \oplus \mathbf{D}_8^{(3)} = \mathbf{I}_{16} \oplus \hat{\mathbf{D}}_8^{(1)} = \left[\begin{array}{c:c:c} \mathbf{I}_8 & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{I}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{D}}_8^{(1)} \end{array}\right].$$

The matrix shuffling the data can be written as:

$$\mathbf{\Gamma}_{24} = \hat{\mathbf{\Gamma}}_8^{(1)} \oplus \hat{\mathbf{\Gamma}}_8^{(2)} \oplus \hat{\mathbf{\Gamma}}_8^{(3)} = \left[\begin{array}{c:c:c} \hat{\mathbf{\Gamma}}_8^{(1)} & \mathbf{0}_8 & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \hat{\mathbf{\Gamma}}_8^{(2)} & \mathbf{0}_8 \\ \hdashline \mathbf{0}_8 & \mathbf{0}_8 & \hat{\mathbf{\Gamma}}_8^{(3)} \end{array}\right],$$

where:

$$\hat{\mathbf{\Gamma}}_8^{(1)} = \mathbf{I}_4 \otimes \mathbf{T}_2^{(1)},\ \hat{\mathbf{\Gamma}}_8^{(2)} = \mathbf{I}_2 \otimes \mathbf{T}_4^{(2)},\ \hat{\mathbf{\Gamma}}_8^{(3)} = \mathbf{T}_8^{(3)},$$

for:

$$\mathbf{T}_2^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \mathbf{T}_4^{(2)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{T}_8^{(3)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The graph-structural model for realization of proposed algorithm for this example is illustrated in figure 1. The circles in this figure show the operation of multiplication by a complex number inscribed inside a circle. In turn, the rectangles indicate the matrix-vector multiplications with the 2×2 Hadamard matrices. In this paper the graph-structural model is oriented from left to right. Straight lines in the figure denote the operation of data transfer. We use the solid lines without any arrows, so as not to clutter up the presented model.

The analysis of computational complexity

Presented algorithm for the calculation of the multiresolution discrete Fourier transform, in accordance with formula (3), requires a following number of complex multiplications:

$$\theta_{\times}^{(i)} = \sum_{j=0}^{m-1} (2^{m-j-1} \cdot 2^j) = (m-i+1) \cdot 2^{m-1},$$

performed in the i-th iteration. Therefore, a total number of complex multiplications performed in the algorithm, can be estimated by summing $\theta_{\times}^{(i)}$ over all iterations:

$$\theta_{\times} = \sum_{i=1}^{m} (m-i+1) \cdot 2^{m-1} = m(m+1) \cdot 2^{m-2}.$$

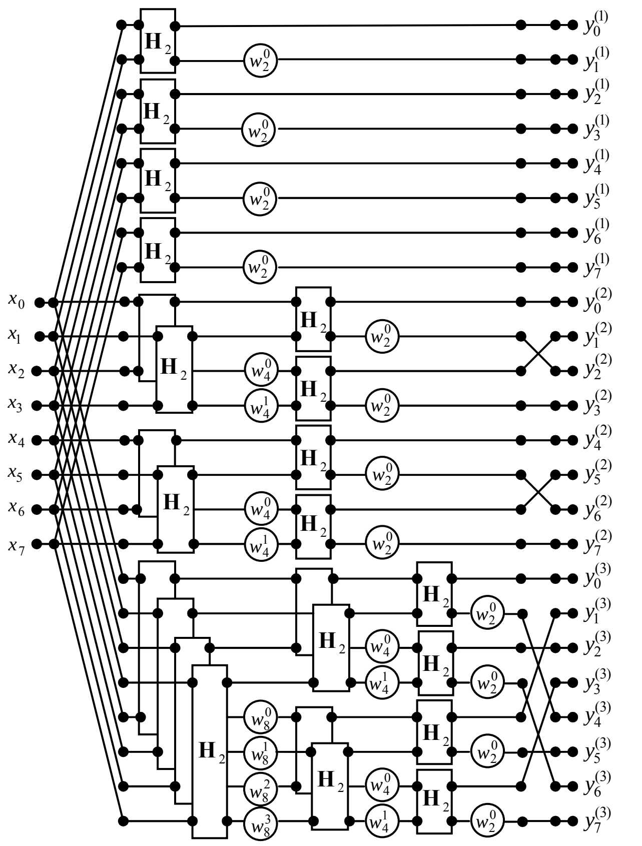
Figure 1. The graph-structural model for computational process organization for column vector $\mathbf{Y}_{24\times1}$ according to procedure (3)

Taking intro account the fact that the described algorithm is based on the Fast Fourier Transform and analyzing the procedure defined by formula (3), it can be noticed that the number of additions performed in each of the iterations is two times bigger than a number of multiplications in the same iteration. Figure 2 presents the relationship between the number of complex multiplications and current iteration for various size of a signal. In turn, figure 3 presents a comparison between total number of complex multiplications performed in according to formula (1) and accordance with formula (3).

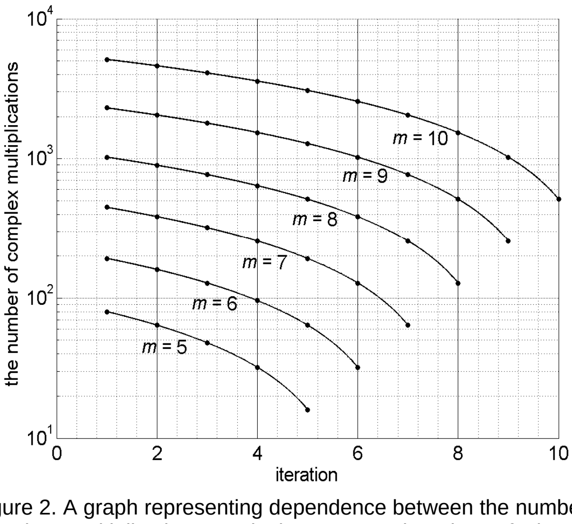

Figure 2. A graph representing dependence between the number of complex multiplications and the current iteration of the fast algorithm

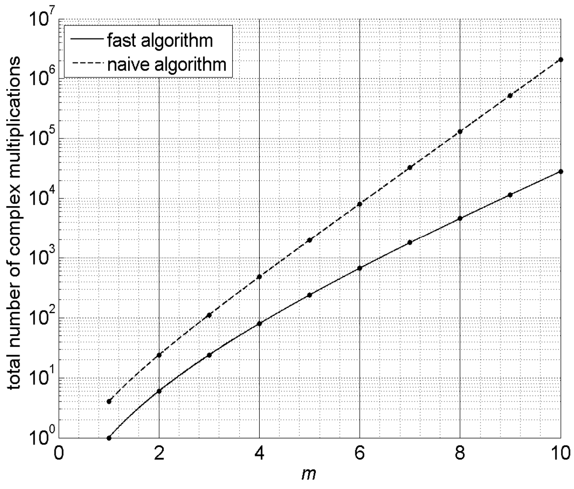

Figure 3. A graph representing comparison between the fast algorithm and the naive method for the MR DFT

Conclusion

In this article, we considered the possibility of reducing the number of calculations performed during realization of the multiresolution discrete Fourier transform. It has been shown that using the FFT algorithm during calculation of spectra of selected segments, we can get the advantage of a significantly lower multiplicative complexity and additive complexity. Moreover, additional advantage of the proposed algorithm is possibility of parallelization. The main drawback of the proposed approach is requirement for the size of segments of the signal, which should be 2^n, where n is some natural number, although this problem is rare in practice. Another disadvantage is the fact that presented solution of the problem do not reuse the internal results.

REFERENCES

[1] Wilson R., Calway A. D., Pearson E. R. S., A Generalized Wavelet Transform for Fourier Analysis: the Multiresolution Fourier Transform and its Application to Image and Audio Signal Analysis., IEEE Transactions on Information Theory, 1992, 38 (2). pp. 674–690.

[2] Annabi-Elkadri N., Hamouda A., Bsaies K., Multi-Resolution Spectral Analysis of Vowels in Tunisian Context, Speech Enhancement, Modeling and Recognition – Algorithms and Applications, in: Speech Processing, K. Ramakrishnan, D. S. (Ed.), 2011, pp. 51-62.

[3] Cancela P., Rocamora M., López E., An Efficient Multi-Resolution Spectral Transform For Music Analysis, 10th International Society for Music Information Retrieval Conference (ISMIR 2009), pp. 309-314.

[4] Dressler K., Sinusoidal Extraction Using An Efficient Implementation of a Multi-Resolution FFT, Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06), Montreal, Canada, September 18-20, 2006, pp. 247-252.

[5] Wen X., Sandler M., Calculation of radix-2 discrete multiresolution Fourier transform, Signal Processing, 2007, v. 87, Issue 10, , Pages 2455–2460.

[6] Blahut R.E., Fast algorithms for digital signal processing, Addison-Wesley Publishing company, Inc. 1985.

[7] Regalia P. A. and Mitra K. S., Kronecker Products, Unitary Matrices and Signal Processing Applications, *SIAM Review*. 1989, v. 31, no. 4, pp. 586-613.

***Autorzy**: inż. Bartosz Andreatto, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, Wydział Informatyki, ul. Żołnierska 49, 70-210 Szczecin, E-mail: bandreatto@wi.ps.pl, dr hab. inż., prof. nadzwyczajny Aleksander Cariow, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, Wydział Informatyki, ul. Żołnierska 49, 70-210 Szczecin, E-mail: atariov@wi.ps.pl*