Aleksandr CARIOW, Galina CARIOWA

West Pomeranian University of Technology, Szczecin

Low complexity algorithm for multiplying octonions

***Abstract**. We propose an original algorithmic solution for multiplication of octonions. In previously published algorithms for computing the product of octonions the number of multiplications has been reduced by significantly increasing number of additions and shifts. A dignity of the proposed solutions is to reduce by 25% the number of multiplications needed to calculate the product of octonions compared with naive method. At the same time the number of additions is the same as in the naive way of calculations. During synthesis of the discussed algorithm we use a fact that octonion product may be represented as a matrix-vector product. Such representation provides a possibility to discover repeating elements in the matrix structure and to use specific properties of their mutual placement for reducing the number of real multiplications needed to calculate the octonion product.*

***Streszczenie.** W artykule przedstawiono szybki algorytm wyznaczania iloczynu oktonionów. Algorytm ten cechuje się zredukowaną o 25% liczbą operacji mnożenia w porównaniu do algorytmu naiwnego przy zachowaniu takiej samej liczby dodawań liczb rzeczywistych. (**Zracjonalizowany algorytm mnożenia oktonionów**).*

Słowa kluczowe: mnożenie oktonionów, oktoniony, szybki algorytm, notacja macierzowa.
Keywords: octonions multiplication, octonions, fast algorithm, matrix notation

doi:10.12915/pe.2014.02.29

Introduction

Currently arithmetic of hypercomplex numbers [1] is increasingly being used to enhance the effectiveness of the solution of problems in various areas of science and technology. Hypercomplex numerical systems are useful in electrodynamics [2], cryptography [3], digital signal and image processing [4, 5], machine graphics [6] and wireless data communications [7]. It should be noted that in the implementation of numerical algorithms using hypercomplex representation of the data, the multiplication is the most time-consuming and labor-intensive. This is because the multiplication of two N dimensional hypercomplex numbers requires N^2 real multiplications and $N(N-1)$ real additions. As can be seen, the complexity of this multiplication is proportional to the square of its dimension. This situation leads to an unacceptable increase in the duration of the operation, or to an increase in hardware costs in case of a VLSI implementation. Therefore, finding ways to reduce the complexity of hypercomplex multiplication is an important task. During the synthesis of efficient algorithms should keep in mind two aspects of design. In low-power digital design, optimization must be done at both algorithmic and logic-circuit levels. Multiplying of hypercomplex numbers involves a large number of real multiplications, which requires much more intensive hardware than real addition operations. Therefore in this case, the hypercomplex numbers product algorithm which contains as little as possible of real multiplications is preferable. On the other hand, when we are dealing with a CPU that contains embedded multipliers, minimizing the number of multiplications by substantially increasing the number of additions is irrational. This is because the delay of addition and multiplication in this case are roughly the same. In this case it is advisable to minimize the total number of arithmetic operations. We should note that first efficient hypercomplex number multiplication algorithms have been developed and published relatively long time ago. Papers [8, 9] describe octonion multiplication algorithms that require about twice less real number multiplications if compared to a direct (naïve) way of doing the calculation. The cost of such decreased number of real number multiplications is almost triple increase of addition and shift operations on real numbers. As a result, the total computational complexity was higher than in the naive method of computing.

This paper aims to present an alternative version of octonion multiplication algorithm that requires performing much less addition on real numbers if compared to the algorithm [9] at the cost of insignificantly increased number of multiplications. Furthermore, the proposed algorithm is completely missing the shift operation (division or multiplication by a power of two.) In this case the total computational complexity of the suggested in this paper algorithm is less than that of the compared solutions.

Formulation of the problem

Suppose we need to compute the product of two octonions:

(1) $$c = ab$$

at that

$$a = (a_0 + ia_1 + ja_2 + ka_3 + Ea_4 + Ia_5 + Ja_6 + Ka_7),$$

$$b = (b_0 + ib_1 + jb_2 + kb_3 + Eb_4 + Ib_5 + Jb_6 + Kb_7),$$

$$c = (c_0 + ic_1 + jc_2 + kc_3 + Ec_4 + Ic_5 + Jc_6 + Kc_7),$$

where $\{a_i\}$, $\{b_i\}$, $\{c_i\}$ $i = 0,1,...,7$, are real numbers and i, j, k, E, I, J, K - are imaginary units.

Table 1 shows multiplication table of imaginary units of octonions [8].

Table 1

$\times$	1	i	j	k	E	I	J	K
1	1	i	j	k	E	I	J	K
i	i	-1	k	$-j$	$-I$	E	K	$-J$
j	j	$-k$	-1	i	$-J$	$-K$	E	I
k	k	j	$-i$	-1	$-K$	J	$-I$	E
E	E	I	J	K	-1	$-i$	$-j$	$-k$
I	I	$-E$	K	$-J$	i	-1	K	$-J$
J	J	$-K$	$-E$	I	j	$-K$	-1	I
K	K	J	$-I$	$-E$	K	J	$-I$	-1

,

In vector-matrix form we can write:

(2) $$\mathbf{Y}_{8\times1}=\mathbf{B}_8\mathbf{X}_{8\times1},$$
where
$$\mathbf{Y}_{8\times1}=[y_0,y_1,y_2,y_3,y_4,y_5,y_6,y_7]^{\mathrm{T}},$$
$$\mathbf{X}_{8\times1}=[x_0,x_1,x_2,x_3,x_4,x_5,x_6,x_7]^{\mathrm{T}},$$

$x_m=a_m,\ y_m=c_m,\ m=0,1,2,3,4,5,6,7,$

$$\mathbf{B}_8=\begin{bmatrix} b_0 & -b_1 & -b_2 & -b_3 & -b_4 & -b_5 & -b_6 & -b_7\\ b_1 & b_0 & b_3 & -b_2 & b_5 & -b_4 & -b_7 & b_6\\ b_2 & -b_3 & b_0 & b_1 & b_6 & b_7 & -b_4 & -b_5\\ b_3 & b_2 & -b_1 & b_0 & b_7 & -b_6 & b_5 & -b_4\\ b_4 & -b_5 & -b_6 & -b_7 & b_0 & b_1 & b_2 & b_3\\ b_5 & b_4 & -b_7 & b_6 & -b_1 & b_0 & -b_3 & b_2\\ b_6 & b_7 & b_4 & -b_5 & -b_2 & b_3 & b_0 & -b_1\\ b_7 & -b_6 & b_5 & b_4 & -b_3 & -b_2 & b_1 & b_0 \end{bmatrix}.$$

The direct multiplication of the vector-matrix product in equation (2) requires 64 real multiplications and 56 real additions. Let us try to explore the possibility of reducing the computational complexity of the implementation of this expression.

Development of the rationalized algorithm for calculating product of octonions

Let us divide matrix into four blocks, each of which contains 8 rows and 2 columns:

$$\mathbf{B}_{8\times2}^{(0)}=\begin{bmatrix} b_0 & -b_1\\ b_1 & b_0\\ b_2 & -b_3\\ b_3 & b_2\\ b_4 & -b_5\\ b_5 & b_4\\ b_6 & b_7\\ b_7 & -b_6 \end{bmatrix},\quad \mathbf{B}_{8\times2}^{(1)}=\begin{bmatrix} -b_2 & -b_3\\ b_3 & -b_2\\ b_0 & b_1\\ -b_1 & b_0\\ -b_6 & -b_7\\ -b_7 & b_6\\ b_4 & -b_5\\ b_5 & b_4 \end{bmatrix},$$

$$\mathbf{B}_{8\times2}^{(2)}=\begin{bmatrix} -b_4 & -b_5\\ b_5 & -b_4\\ b_6 & b_7\\ b_7 & -b_6\\ b_0 & b_1\\ -b_1 & b_0\\ -b_2 & b_3\\ -b_3 & -b_2 \end{bmatrix},\quad \mathbf{B}_{8\times2}^{(3)}=\begin{bmatrix} -b_6 & -b_7\\ -b_7 & b_6\\ -b_4 & -b_5\\ b_5 & -b_4\\ b_2 & b_3\\ -b_3 & b_2\\ b_0 & -b_1\\ b_1 & b_0 \end{bmatrix}.$$

Then we can write:

(3) $$\mathbf{Y}_{8\times1}=\mathbf{B}_8\mathbf{X}_{8\times1}=\sum_{n=0}^{3}(\mathbf{B}_{8\times2}^{(n)}\mathbf{X}_{2\times1}^{(n)}),$$

where
$$\mathbf{X}_{2\times1}^{(0)}=[x_0,x_1]^{\mathrm{T}},\ \mathbf{X}_{2\times1}^{(1)}=[x_2,x_3]^{\mathrm{T}},$$
$$\mathbf{X}_{2\times1}^{(2)}=[x_4,x_5]^{\mathrm{T}},\ \mathbf{X}_{2\times1}^{(3)}=[x_6,x_7]^{\mathrm{T}}.$$

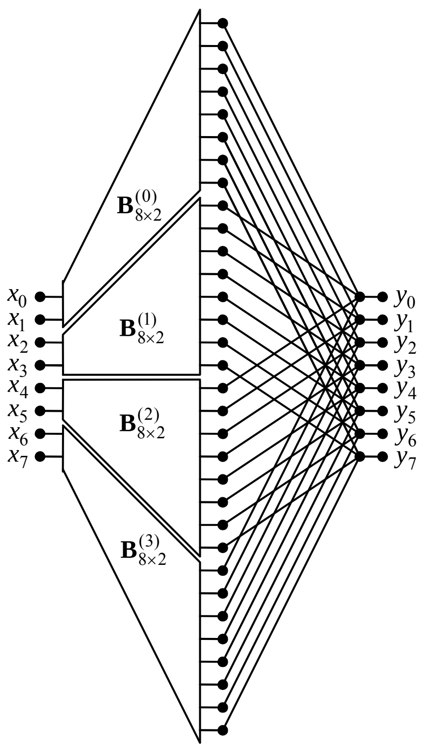
In more acceptable form (3) can be rewritten as:

(4) $$\mathbf{Y}_{8\times1}=\mathbf{A}_{8\times32}\left(\bigoplus_{n=0}^{3}\mathbf{B}_{8\times2}^{(n)}\right)\mathbf{X}_{8\times1},$$

where
$$\mathbf{A}_{8\times32}=\mathbf{1}_{1\times4}\otimes\mathbf{I}_8,$$

and $\mathbf{1}_{m\times n}$ denotes the matrix whose elements are all units. $\mathbf{I}_N$ is an identity $N\times N$ matrix, in turn the symbols $\otimes$ and $\oplus$ indicate a Kronecker product and a direct sum of two matrices respectively.

Figure 1, in order to facilitate an understanding of the main idea of the transformation, shows the data-flow diagram of the octonions product calculation in accordance with procedure given by (4). In this article all data-flow diagrams are oriented from left to right. Straight lines in the figures denote the operation of data transfer. In this paper we use solid lines without any arrows, so as not to clutter up the presented diagrams. At points where lines converge, the data are summarized. The rectangles and circles indicate the operation of multiplication by the matrix or variable inscribed inside an element.



x_0
x_1
x_2
x_3
x_4
x_5
x_6
x_7
$\mathbf{B}_{8\times2}^{(0)}$
$\mathbf{B}_{8\times2}^{(1)}$
$\mathbf{B}_{8\times2}^{(2)}$
$\mathbf{B}_{8\times2}^{(3)}$
y_0
y_1
y_2
y_3
y_4
y_5
y_6
y_7

Figure 1. The graph-structural model for decomposing process of octonions product calculation into four subprocesses according to procedure (3)

As can be seen from Figure 1, the computation process for multiplication of matrix $\mathbf{Q}_8$ by vector $\mathbf{X}_{8\times1}$ can be implemented as four independent vector-matrix products $\mathbf{B}_{8\times2}^{(n)}\mathbf{X}_{2\times1}^{(n)}$. The results of these calculations are later be added together. It should be noted that the ordinary implementation of (4) does not provide any reduction in computational complexity. However, we will notice that the sub blocks of the matrices $\mathbf{B}_{8\times2}^{(n)}$ separated by the dashed lines possess specific structures. This specificity, as we show below, allows reducing the number of multiplications in the implementation of the partials vector-matrix products. The mentioned possibility of rationalization uses the following decomposition [10]:

$$\left[\begin{array}{c|c} a & b \\ \hline c & a \end{array}\right] = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \left[\begin{array}{c:c:c} c-a & 0 & 0 \\ \hdashline 0 & b-a & 0 \\ \hdashline 0 & 0 & a \end{array}\right] \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

As can be seen, each of such vector-matrix multiplications require only three multiplications and five additions. Using these singularities can offer a more cost-effective way to obtain a set of partial vector-matrix products:

(5) $$\mathbf{Y}^{(n)}_{12\times1} = \mathbf{P}^{(n)}_{12}\mathbf{D}^{(n)}_{12}\mathbf{P}^{(n)}_{12\times9}\mathbf{A}_{9\times10}\mathbf{P}_{10\times2}\mathbf{X}^{(n)}_{2\times1},$$

where

$$\mathbf{Y}^{(n)}_{12\times1} = [y^{(n)}_0, y^{(n)}_1, ..., y^{(n)}_{11}]^{\mathrm{T}},$$

$$\mathbf{P}_{10\times2} = \mathbf{1}_{5\times1} \otimes \mathbf{I}_2, \ \mathbf{A}_{9\times10} = \mathbf{I}_8 \oplus \mathbf{1}_{1\times2},$$

$$\mathbf{P}^{(0)}_{12\times9} = \mathbf{I}_8 \oplus \mathbf{1}_{4\times1}, \ \mathbf{P}^{(1)}_{12\times9} = (\mathbf{I}_2 \otimes \mathbf{J}_2) \oplus \mathbf{I}_4 \oplus \mathbf{1}_{4\times1},$$

$$\mathbf{P}^{(2)}_{12\times9} = (\mathbf{I}_2 \otimes (\mathbf{J}_2 \oplus \mathbf{I}_2)) \oplus \mathbf{1}_{4\times1},$$

$$\mathbf{P}^{(3)}_{12\times9} = \mathbf{I}_2 \oplus (\mathbf{I}_2 \otimes \mathbf{J}_2) \oplus \mathbf{I}_2 \oplus \mathbf{1}_{4\times1},$$

$$\mathbf{P}^{(0)}_{12} = (\mathbf{I}_2 \otimes \mathbf{J}_2) \oplus \mathbf{I}_6, \ \mathbf{P}^{(1)}_{12} = \mathbf{P}^{(2)}_{12} = \mathbf{P}^{(3)}_{12} = (\mathbf{I}_6 \oplus \mathbf{J}_2 \oplus \mathbf{I}_4),$$

$$\mathbf{D}^{(n)}_{12} = diag(s^{(n)}_0, s^{(n)}_1, ..., s^{(n)}_{11}), \ n = 0,1,2,3,$$

$$\mathbf{D}^{(0)}_{12} = diag(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, b_0, b_2, b_4, b_7),$$

$$\mathbf{D}^{(1)}_{12} = diag(-s_2, -s_3, s_0, s_1, -s_6, -s_7, s_4, s_5, -b_2, b_0, -b_7, b_4),$$

$$\mathbf{D}^{(2)}_{12} = diag(-s_4, -s_5, s_6, s_7, s_0, s_1, -s_2, -s_3, -b_4, b_7, b_0, -b_2),$$

$$\mathbf{D}^{(3)}_{12} = diag(-s_6, -s_7, -s_4, -s_5, s_2, s_3, s_0, s_1, -b_7, -b_4, b_2, -b_0).$$

Generalized data-flow diagram in Figure 3 illustrates the organization of computing one (each) of the four vector-matrix products, described by the expression 5. It is easy to see that the elements $\{s_i\}$ of the matrices $\mathbf{D}^{(n)}_{12}$ can be calculated using the following vector–matrix procedure:

(6) $$\mathbf{S}_{8\times1} = \hat{\mathbf{J}}_8\hat{\mathbf{D}}_8\hat{\mathbf{H}}_8\hat{\mathbf{J}}_8\mathbf{Q}_{8\times1}$$

where:

$$\mathbf{Q}_{8\times1} = [b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7]^{\mathrm{T}},$$

$$\mathbf{S}_{8\times1} = [s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7]^{\mathrm{T}},$$

$$\hat{\mathbf{J}}_8 = \mathbf{I}_4 \otimes \mathbf{J}_2, \ \hat{\mathbf{H}}_8 = \mathbf{I}_4 \otimes \mathbf{H}_2,$$

$$\hat{\mathbf{D}}_8 = diag(-1,1,-1,1,-1,1,-1,1),$$

$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ is the 2×2 Hadamard matrix,

and $\mathbf{J}_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ - is a (2×2) exchange matrix,

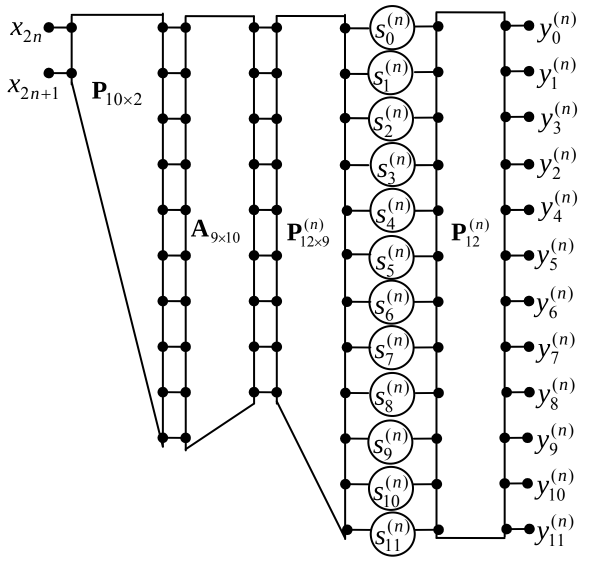


Figure 2. The graph-structural model for computational process organization for calculating of column vector $\mathbf{Y}^{(n)}_{12\times1}$ according to procedure (5)

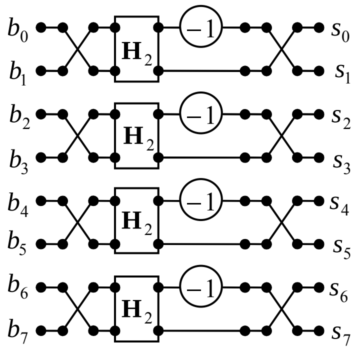


Figure3. The graph-structural model of computation process organization for vector $\mathbf{S}_{8\times1}$ elements calculating corresponding to procedure (6)

Let us present the results of calculations of the partial vector-matrix products as one large vector

$$\mathbf{Y}_{48\times1} = [\mathbf{Y}^{(0)}_{12\times1}, \mathbf{Y}^{(1)}_{12\times1}, \mathbf{Y}^{(2)}_{12\times1}, \mathbf{Y}^{(3)}_{12\times1}]^{\mathrm{T}}.$$

Then the procedure for calculating the elements of this vector can be represented in the following form.

(7) $$\mathbf{Y}_{48\times1} = \mathbf{P}'_{48}\mathbf{D}_{48}\mathbf{P}_{48\times36}\mathbf{A}_{36\times40}\mathbf{P}_{40\times8}\mathbf{X}_{8\times1},$$

where

$$\mathbf{P}_{40\times8} = \mathbf{I}_4 \otimes \mathbf{P}_{10\times2}, \ \mathbf{A}_{36\times40} = \mathbf{I}_4 \otimes \mathbf{A}_{9\times10}$$

$$\mathbf{P}_{48\times36} = \mathbf{P}^{(0)}_{12\times9} \oplus \mathbf{P}^{(1)}_{12\times9} \oplus \mathbf{P}^{(2)}_{12\times9} \oplus \mathbf{P}^{(3)}_{12\times9},$$

$$\mathbf{D}_{48} = \mathbf{D}^{(0)}_{12} \oplus \mathbf{D}^{(1)}_{12} \oplus \mathbf{D}^{(2)}_{12} \oplus \mathbf{D}^{(3)}_{12},$$

$$\mathbf{P}'_{48} = \mathbf{P}_{12}^{(0)} \oplus \mathbf{P}_{12}^{(1)} \oplus \mathbf{P}_{12}^{(2)}\mathbf{P}_{12}^{(3)},$$

If all the operations related to the implementation of computing the vector-matrix products in accordance to (5) and (7) are made, the result of multiplying two octonions can be obtained as follows (see Fig. 4):

$$\mathbf{Y}_{8\times1} = \mathbf{A}_{8\times12}\mathbf{A}_{12\times48}\mathbf{P}''_{48}\mathbf{Y}_{48\times1}, \quad (8)$$

$$\mathbf{A}_{12\times48} = \mathbf{I}_{12} \otimes \mathbf{1}_{1\times4}, \ \mathbf{A}_{8\times12} = \mathbf{I}_4 \otimes \mathbf{T}_{2\times3}, \ \mathbf{T}_{2\times3} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix},$$

$\mathbf{P}''_{48} = \mathop{\Big|}\limits_{l=0}^{N-1} (\mathbf{I}_4 \otimes \mathbf{e}_{1\times12}^{(l)})$, where $\mathbf{e}_{1\times12}^{(l)} = [\underbrace{0,0,\ldots,1}_{l},..0,0,\ldots0]$ - is the l-th row of 12×12 identity matrix, $l = 0,1,\ldots,11$, and $\Big|$ denotes horizontal concatenation sign [11].

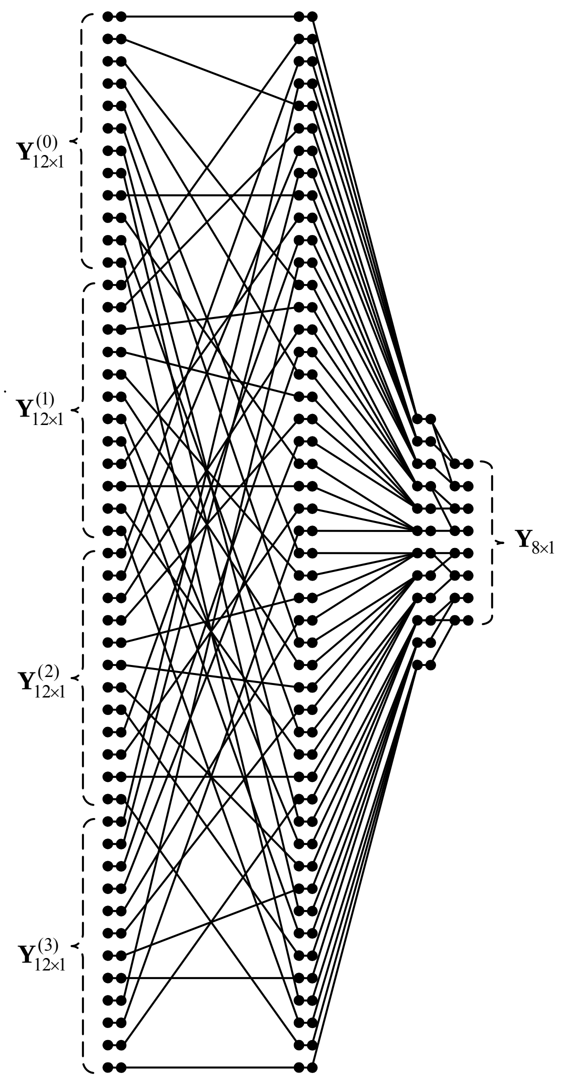

$\mathbf{Y}_{12\times1}^{(0)}$
$\mathbf{Y}_{12\times1}^{(1)}$
$\mathbf{Y}_{12\times1}^{(2)}$
$\mathbf{Y}_{12\times1}^{(3)}$
$\mathbf{Y}_{8\times1}$

Figure 4. The graph-structural model for computational process organization for column vector $\mathbf{Y}_{48\times1}$ according to procedure (8)

The number of real multiplications required using the proposed algorithm is 48. Thus using the proposed algorithm the number of real multiplications to implement the octonions product is reduced. The number of real additions required using our algorithm is 56. We observe that the proposed algorithm requires 16 multiplications less, than the direct computation of octonion product while maintaining the same number of additions. Nonetheless, the total number of arithmetic operations for proposed algorithm is approximately 15% less than that of the direct evaluation. Table 2 shows the estimates of the computational complexity of algorithms for multiplication of octonions.

Table 2 Evaluation of the computational complexity of algorithms

Solution	Number of multiplications	Number of adds	Number of shifts	Total arithmetical complexity
Naïve (Eq.1)	64	56	—	120
Algorithm [8]	32	88	16	136
Algorithm [9]	30	94	32	156
Proposed	48	56	—	104

Conclusion

We presented a new algorithm for calculating the product of two octonions. The use of this algorithm reduces the computational complexity of multiplications of octonions. We state that this algorithm has better computational complexity than algorithm described in [9], even though is contains 18 multiplication more. Additionally, we note that the total number of arithmetic operations in our algorithm is less than the total number of operations in the compared algorithms. Therefore, the proposed algorithm is better than the algorithms [8, 9], even in terms of its software implementation on a conventional computer. In some cases the presented algorithm may appear to be more applicable and convenient from the implementation point of view.

REFERENCES

[1] Kantor, I.L. and Solodovnikov A. S. "Hypercomplex numbers. An elementary introduction to algebras". Springer: New York, 1989.
[2] Chanyal B. C., Bisht P. S. and Negi O. P. S., "Generalized Octonion Electrodynamics", Int. J. Theor. Phys., 49 (2010), 137.
[3] Malekian E., Zakerolhosseini A., Mashatan A., QTRU: Quaternionic Version of the NTRU Public-Key Cryptosystems, Int. J. Inf. Secur., 3, 29-42, 2011
[4] Alfsmann D., Göckler H. G., Sangwine S. J. and Ell T. A. Hypercomplex Algebras in Digital Signal Processing: Benefits and Drawbacks (Tutorial). Proc. EURASIP 15th European Signal Processing Conference (EUSIPCO 2007), Poznań, Poland, 2007, 1322-1326.
[5] Moxey C.E., Sangwine S. J., Ell T.A., Hypercomplex correlation techniques for vector images, IEEE Trans. Signal Process., 2003. 51, 1941-1953
[6] Wang Hui; Wang Xiao-Hui; Zhou Yue; Yang Jie; "Color Texture Segmentation Using Quaternion-Gabor Filters", Image Processing, 2006 IEEE International Conference on. 8-11 Oct. 2006, 745 – 748.
[7] Calderbank R., Das S., Al Dhahir N., Diggavi S., Construction And Analysis of A New Quaternionic Space-Time Code For 4 Transmit Antennas, Commun. Inf. Syst., 5, 97-122, 2005
[8] Cariow A., Cariowa G., Algorithm for multiplying two octonions, Radioelectronics and Communications Systems (Allerton Press, Inc. USA), October 2012, Volume 55, Issue 10, pp 464-473, ISSN 0735-2727.
[9] Ţariov A., Ţariova G., Aspekty algorytmiczne organizacji układu procesorowego do mnożenia liczb Cayleya. Elektronika, No 11, 2010, 137-140
[10] Ţariov A., Strategie racjonalizacji obliczeń przy wyznaczaniu iloczynów macierzowo-wektorowych. Metody Informatyki Stosowanej, Nr 1, 2008, 147- 158.
[11] Ţariov A., Algorytmiczne aspekty racjonalizacji obliczeń w cyfrowym przetwarzaniu sygnałów, Wydawnictwo Uczelniane ZUT, 2011. - 232 s.

Autorzy: dr hab. inż. Aleksandr Cariow, prof. nadzwyczajny Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, Wydział Informatyki, ul. Żołnierska 49, 70-210 Szczecin, E-mail: atariov@wi.zut.edu.pl; dr Galina Cariowa, adiunkt, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, Wydział Informatyki, ul. Żołnierska 49, 70-210 Szczecin, E-mail: gtariova@wi.zut.edu.pl