THE COMPLEXITY OF THE QUATERNION PRODUCT*

Thomas D. Howell

Jean-Claude Lafon[1]**

TR 75-245

June 1975[2]

Department of Computer Science, Cornell University, Ithaca, N.Y.

* This research was supported in part by the National Science Foundation Graduate Fellowship and the Office of Naval Research Grant N00014-67-A-0077-0021.

** Universite Scientifique et Medicale de Grenoble, Tour des Mathematiques, B.P. 53, 38041 Grenoble, France.

Abstract:

Let X and Y be two quaternions over an arbitrary ring. Eight multiplications are necessary and sufficient for computing the product XY. If the ring is assumed to be commutative, at least seven multiplications are still necessary and eight are sufficient.

1. Introduction

Real quaternions are elements of the division ring over the reals generated by the elements $\mathbf{i},\mathbf{j},\mathbf{k}$ satisfying $\mathbf{i}^2=\mathbf{j}^2=\mathbf{k}^2=-1$, $\mathbf{ij}=-\mathbf{ji}=\mathbf{k}$, $\mathbf{jk}=-\mathbf{kj}=\mathbf{i}$ and $\mathbf{ki}=-\mathbf{ik}=\mathbf{j}$. If $x_n, y_n \in R$ for $n=1,2,3,4$, then $X=x_1+x_2\mathbf{i}+x_3\mathbf{j}+x_4\mathbf{k}$ and $Y=y_1+y_2\mathbf{i}+y_3\mathbf{j}+y_4\mathbf{k}$ are quaternions and their product is:

$$\begin{aligned} XY = {} & (x_1y_1 - x_2y_2 - x_3y_3 - x_4y_4) \\ & + (x_1y_2 + x_2y_1 + x_3y_4 - x_4y_3)\,\mathbf{i} \\ & + (x_1y_3 - x_2y_4 + x_3y_1 + x_4y_2)\,\mathbf{j} \\ & + (x_1y_4 + x_2y_3 - x_3y_2 + x_4y_1)\,\mathbf{k} \end{aligned}$$

Real quaternions satisfy $\mathrm{N}(X) \equiv x_1^2+x_2^2+x_3^2+x_4^2 > 0$ if $X \neq 0$ and $\mathrm{N}(XY) = \mathrm{N}(X)\,\mathrm{N}(Y)$. They are important because of their usefulness in calculations arising in physics and mathematics.

---

[1]Now at the University of Nice (`jcl@mimosa.unice.fr`)

[2]Retyped and converted to TeX from a TIFF G4 600dpi scanned image of the original obtained from the Cornell Computer Science WWW site (also now stored at `ftp://ftp.netcom.com/pub/hb/hbaker/quaternion/cornellcstr75-245.tar.gz`), November, 1995, by Henry G. Baker (`hbaker@netcom.com`), and posted on the Internet (`ftp://ftp.netcom.com/pub/hb/hbaker/quaternion/cornellcstr75-245.dvi.gz` and `.ps.gz`) with the permission of Jean-Claude Lafon.

Although real quaternions are most often used, the quaternion product as defined above can be computed when the x_n's and y_n's are elements of an arbitrary ring. We are interested in the complexity of this computation. The complexity of an algorithm is measured by the number of ring multiplications it requires. The obvious algorithm for computing the quaternion product uses 16 multiplications. The complexity of the quaternion product is the complexity of the least complex algorithm computing it.

Two cases must be considered when discussing the complexity of quaternion multiplication over a ring, the general case and the commutative case. The general case considers only those algorithms which correctly compute the quaternion product over any ring.

The commutative case allows, in addition to these, algorithms which work only in commutative rings. Commutative algorithms are important because the real numbers are commutative. Clearly the complexity of the quaternion product, or any computation, in the commutative case is not greater than the complexity of the same computation in the general case.

The problem of determining the complexity of the quaternion product is a special case of the problem of determining the complexity of a set of bilinear forms. A number of other special cases appear in the literature. For example, Strassen [1] shows that seven multiplications are sufficient for the product of 2×2 matrices. Hopcroft and Kerr [2] and Winograd [3] show that seven multiplications are also necessary in the general and commutative cases, respectively. Fiduccia [4] mentions that ten multiplications are sufficient for the quaternion product. It has been shown by de Groote that ten are necessary and sufficient to compute the quaternion products XY and YX simultaneously. The complexity of arbitrary sets of bilinear forms has been studied by Gastinel [6] and Musinski [7], but useful results are known only for sets of size one or two.

The purpose of this paper is to show that eight multiplications are necessary and sufficient for the quaternion product in the general case, and that at least seven multiplications are necessary in the commutative case.

2. Model of Computation

The following definitions make precise the notion of an algorithm and its complexity.

Definition. Let F be a field and $\{x_1, x_2, ..., x_n\}$ be a set of indeterminates. An *algorithm* is a finite sequence of steps, $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_s\}$, where each step, α_i, is of one of three forms:

(i) $g_i \leftarrow x_j, 1 \leq j \leq n$

(ii) $g_i \leftarrow a, a \in F$

(iii) $g_i \leftarrow g_j \theta g_k, j < i, k < i$ and $\theta \in \{+, -, \times\}$.

Note that each g_i is an element of $F[x_1, ..., x_n]$, the ring of polynomials in $x_1, ..., x_n$ with coefficients from F. An algorithm α is said to compute a set $E \subseteq F[x_1, ..., x_n]$ if $E \subseteq \{g_1, ..., g_s\}$.

Definition. The *complexity* of an algorithm is equal to the number of steps of the form $g_i \leftarrow g_j \times g_k$ where $g_j \notin F$ and $g_k \notin F$. Such steps are called *active multiplications*.

The x_i's are the input data to the algorithm. Only multiplications which depend on the data are counted. Multiplications by scalars (elements of F) are ignored.

The complexity of a computation may depend on the choice of F, the field of scalars. For example, $x_1^2 + x_2^2$ requires two active multiplications when F is the reals. When F is the complex numbers, $x_1^2 + x_2^2 = (x_1 + ix_2)(x_1 - ix_2)$ requires just one active multiplication. Throughout this paper F is assumed to satisfy the following condition: For all $a, b, c, d, \in F$

$$a^2 + b^2 + c^2 + d^2 = 0 \quad \text{if and only if} \quad a = b = c = d = 0. \qquad (*)$$

The reals and the rationals satisfy (*), but the complex numbers and all finite fields do not. A counterexample will show that our theorems do not hold when F is the complexes.

3. Tensor Rank

Let $\{M_k\}_{k=1}^p$ be a set of $m \times n$ matrices over F. Let $x^t = [x_1, ..., x_m]$ and $y^t = [y_1, ..., y_n]$ be the vectors of indeterminates. Let f_k be the bilinear form represented by $M_k : f_k = x^t M_k y$ for $k = 1, ..., p$. The following theorem is known [8], [9].

Theorem 1. If α is an algorithm of complexity q computing $f_1, ..., f_p$ in the general case (commutativity is not assumed) then there is an algorithm α' of complexity q computing $f_1, ..., f_p$ in which all active multiplications are the form $(u_\ell^t x) \cdot (v_\ell^t y)$ where $u_\ell \in F^m$, $v_\ell \in F^n$ for $\ell = 1, 2, ..., q$. Then

$$\begin{bmatrix} f_1 \\ f_2 \\ \cdot \\ \cdot \\ \cdot \\ f_p \end{bmatrix} = W \begin{bmatrix} (u_1^t x) \cdot (v_1^t y) \\ (u_2^t x) \cdot (v_2^t y) \\ \cdot \\ \cdot \\ \cdot \\ (u_q^t x) \cdot (v_q^t y) \end{bmatrix}$$

where W is a $p \times q$ matrix over F.

Let $z^t = [z_1, ..., z_p]$ be another vector of indeterminates.

The trilinear form

$$T = \sum_{k=1}^{p} f_k z_k = \sum_{i,j,k} m_{ijk} x_i y_j z_k$$

is represented by $M = (m_{ijk})_{i=1}^{m}{}_{j=1}^{n}{}_{k=1}^{p} \in F^m \otimes F^n \otimes F^p$. Note that m_{ijk} is the i, j element of M_k.

Definition. The *tensor rank* of the set $\{M_k\}_{k=1}^p$ (or simply of M) is the smallest integer q for which

$$M = \sum_{\ell=1}^{q} u_\ell \otimes v_\ell \otimes w_\ell$$

where $u_\ell \in F^m, v_\ell \in F^n, w_\ell \in F^p$ for $\ell = 1,...,q$, and $\otimes$ denotes tensor product (outer product).

When $p = 1$, the tensor rank of the set $\{M_1\}$ is equal to the rank of the matrix M_1. The next theorem shows the importance of tensor rank.

Theorem 2. The tensor rank of the set $\{M_k\}_{k=1}^p$ is equal to the minimum number of active multiplications required to compute the bilinear forms $f_k = x^t M_k y, k = 1,...,p$, in the general case.

Proof: Assume q active multiplications suffice. By Theorem 1

$$f_k = \sum_{\ell=1}^{q} w_{k\ell}(u_\ell^t x)(v_\ell^t y).$$

Let $w_\ell^t = [w_{1\ell},...,w_{p\ell}]$. Then

$$T = \sum_{i,j,k} m_{ijk} x_i y_j z_k = \sum_{k=1}^{p} f_k z_k = \sum_{\ell=1}^{q} (u_\ell^t x)(v_\ell^t y)(w_\ell^t z),$$

which implies

$$M = \sum_{\ell=1}^{q} u_\ell \otimes v_\ell \otimes w_\ell.$$

Hence the tensor rank of M is at most q. The reverse argument shows that if the tensor rank of M is q then q active multiplications suffice to compute $f_1,...,f_p$.

It will often be more convenient to represent the three-dimensional object M in two dimensions as $M(z) = \sum_{k=1}^{p} M_k z_k$. Then Theorem 2 says that the complexity of the set of bilinear forms $f_1,...,f_p$ is equal to the smallest q such that

$$M(z) = \sum_{\ell=1}^{q} u_\ell \otimes v_\ell(w_\ell^t z) = \sum_{\ell=1}^{q} u_\ell v_\ell^t (w_\ell^t z) \quad ([10]).$$

Thus, q is the minimum number of rank-one matrices (depending on z) into which $M(z)$ can be decomposed. This will be referred to as the tensor rank of $M(z)$.

Example. Let $M_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, M_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Then

$$
\begin{aligned}
M(z) &= \begin{bmatrix} z_1 & z_2 \\ z_2 & -z_1 \end{bmatrix} \\
&= \begin{bmatrix} z_2 & z_2 \\ z_2 & z_2 \end{bmatrix} + \begin{bmatrix} z_1 - z_2 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -z_1 - z_2 \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ 1 \end{bmatrix} [1 \quad 1] (z_2) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} [1 \quad 0] (z_1 - z_2) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} [0 \quad 1] (-z_1 - z_2)
\end{aligned}
$$

hence the tensor rank of $M(z)$ is at most three.

$$
\begin{aligned}
T &= (x_1y_1 - x_2y_2)z_1 + (x_1y_2 + x_2y_1)z_2 \\
&= (x_1 + x_2)(y_1 + y_2)z_2 + x_1y_1(z_1 - z_2) + x_2y_2(-z_1 - z_2)
\end{aligned}
$$

This leads to

$$
\begin{bmatrix} x_1y_1 - x_2y_2 \\ x_1y_2 + x_2y_1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} (x_1 + x_2) \cdot (y_1 + y_2) \\ x_1 \cdot y_1 \\ x_2 \cdot y_2 \end{bmatrix}
$$

which is of the form given in Theorem 1.

For the general case, Theorem 2 has characterized the complexity of a set of bilinear forms in terms of the tensor rank of the coefficients. The same can be done for the commutative case.

Let $M(z)$ be as above.

Theorem 3. The commutative complexity of the bilinear forms $f_1, ..., f_p$ is equal to the minimum possible tensor rank of $N(z)$ where

$$
N(z) + N(z)^t = \begin{bmatrix} 0 & M(z) \\ M(z)^t & 0 \end{bmatrix}.
$$

This follows from a theorem analogous to Theorem 1, which states that in the commutative case there is a minimal complexity algorithm in which the active multiplications have the form $(u_\ell^t x + v_\ell^t y)(\tilde{u}_\ell^t x + \tilde{v}_\ell^t y)$ where $u_\ell, \tilde{u}_\ell \in R^m$ and $v_\ell, \tilde{v}_\ell \in R^n$. See Lafon [9] for a proof.

Note that the choice

$$
N(z) = \begin{bmatrix} 0 & M(z) \\ 0 & 0 \end{bmatrix}
$$

is possible, hence the commutative complexity is not greater than the general complexity of a set of bilinear forms.

4. Upper Bound

In this section an algorithm will be given which computes the quaternion product in the general case using eight active multiplications.

The quaternion product consists of four bilinear forms:

$$\begin{aligned}
f_1 &= x^t M_1 y = x_1y_1 - x_2y_2 - x_3y_3 - x_4y_4\\
f_2 &= x^t M_2 y = x_1y_2 + x_2y_1 + x_3y_4 - x_4y_3\\
f_3 &= x^t M_3 y = x_1y_3 - x_2y_4 + x_3y_1 + x_4y_2\\
f_4 &= x^t M_4 y = x_1y_4 + x_2y_3 - x_3y_2 + x_4y_1
\end{aligned}$$

By the results of section 3, the desired algorithm corresponds to a decomposition of $M(z)$ as a sum of eight rank-one matrices, where

$$N(z) = \sum_{k=1}^{4} M_k z_k = \begin{bmatrix} z_1 & z_2 & z_3 & z_4 \\ z_2 & -z_1 & z_4 & -z_3 \\ z_3 & -z_4 & -z_1 & z_2 \\ z_4 & z_3 & -z_2 & -z_1 \end{bmatrix}.$$

First, write $M(z) = M'(z) + M''(z)$ where

$$M'(z) = \begin{bmatrix} -z_1 & z_2 & z_3 & z_4 \\ z_2 & -z_1 & z_4 & z_3 \\ z_3 & z_4 & -z_1 & z_2 \\ z_4 & z_3 & z_2 & -z_1 \end{bmatrix}$$

$$M''(z) = 2\begin{bmatrix} z_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -z_3 \\ 0 & -z_4 & 0 & 0 \\ 0 & 0 & -z_2 & 0 \end{bmatrix}$$

Obviously, $M''(z)$ is representable as a sum of four rank-one matrices. So is $M'(z)$:

$$4M'(z) =$$
$$(-z_1+z_2+z_3+z_4)\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} + (-z_1+z_2-z_3-z_4)\begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \end{bmatrix} +$$
$$(-z_1-z_2+z_3-z_4)\begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix} + (-z_1-z_2-z_3+z_4)\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

The corresponding algorithm is :

$$[\mathrm{I}] = x_1y_1 \qquad [\mathrm{II}] = x_4y_3 \qquad [\mathrm{III}] = x_2y_4 \qquad [\mathrm{IV}] = x_3y_2$$

$$
\begin{aligned}
[\mathrm{V}] &= (x_1+x_2+x_3+x_4)(y_1+y_2+y_3+y_4)\\
[\mathrm{VI}] &= (x_1+x_2-x_3-x_4)(y_1+y_2-y_3-y_4)\\
[\mathrm{VII}] &= (x_1-x_2+x_3-x_4)(y_1-y_2+y_3-y_4)\\
[\mathrm{VIII}] &= (x_1-x_2-x_3+x_4)(y_1-y_2-y_3+y_4)\\
f_1 &= 2[\mathrm{I}] - \frac{[\mathrm{V}]+[\mathrm{VI}]+[\mathrm{VII}]+[\mathrm{VIII}]}{4}\\
f_2 &= -2[\mathrm{II}] + \frac{[\mathrm{V}]+[\mathrm{VI}]-[\mathrm{VII}]-[\mathrm{VIII}]}{4}\\
f_3 &= -2[\mathrm{III}] + \frac{[\mathrm{V}]-[\mathrm{VI}]+[\mathrm{VII}]-[\mathrm{VIII}]}{4}\\
f_4 &= -2[\mathrm{IV}] + \frac{[\mathrm{V}]-[\mathrm{VI}]-[\mathrm{VII}]+[\mathrm{VIII}]}{4}
\end{aligned}
$$

This shows that the complexity of the quaternion product is at most eight in both the general case and the commutative case.

5. Transformations

An algorithm computing the bilinear forms, $S = \{x^t M_k y : k = 1, \dots, p\}$, can often be transformed into a new algorithm computing the same forms by substituting for each x_i a linear combination of the x_i's and for each y_j a linear combination of the y_j's and then taking new linear combinations of the resulting active multiplications.

Definition. Let A, B, and C be nonsingular matrices over F of order m, n and p, respectively. The ordered triple, (A, B, C), is a *transformation* on the set S if it leaves the trilinear form representing S invariant:

$$\sum_{i,j,k} m_{ijk} (Ax)_i (By)_j (Cz)_k = \sum_{i,j,k} m_{ijk} x_i y_j z_k.$$

The transformations on a given set of forms form a group under composition.

Given an algorithm computing S and a transformation, (A, B, C), a new algorithm of the same complexity can be formed by first computing $x' = Ax$ and $y' = By$ using no active multiplications. The original algorithm applied to x' and y' computes the coefficients $f' = [f'_1, \dots, f'_p]^t$ of $z' = Cz$ in the expression $\sum_{i,j,k} m_{ijk} x'_i y'_j z'_k$. The desired forms, S, which are the coefficients, $f = [f_1, \dots, f_p]$, of z are recovered by computing $f'^t C = f^t$ using no active multiplications.

Example.

Let

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad C = \frac{1}{4} AB$$

The reader can verify by direct computation that (A,B,C) is a transformation on the quaternion product. When applied to the algorithm of section 4, it yields the following algorithm:

$$
\begin{array}{rcl@{\qquad\qquad}rcl}
[\mathrm{I}] & = & (x_1+x_2)(y_1+y_2) & [\mathrm{V}] & = & (x_2+x_4)(y_2+y_3) \\
[\mathrm{II}] & = & (x_4-x_3)(y_3-y_4) & [\mathrm{VI}] & = & (x_2-x_4)(y_2-y_3) \\
[\mathrm{III}] & = & (x_2-x_1)(y_3+y_4) & [\mathrm{VII}] & = & (x_1+x_3)(y_1-y_4) \\
[\mathrm{IV}] & = & (x_3+x_4)(y_2-y_1) & [\mathrm{VIII}] & = & (x_1-x_3)(y_1+y_4)
\end{array}
$$

$$
\begin{aligned}
f_1 &= [\mathrm{II}] + \frac{-[\mathrm{V}]-[\mathrm{VI}]+[\mathrm{VII}]+[\mathrm{VIII}]}{2} \\
f_2 &= [\mathrm{I}] - \frac{[\mathrm{V}]+[\mathrm{VI}]+[\mathrm{VII}]+[\mathrm{VIII}]}{2} \\
f_3 &= -[\mathrm{III}] + \frac{[\mathrm{V}]-[\mathrm{VI}]+[\mathrm{VII}]-[\mathrm{VIII}]}{2} \\
f_4 &= -[\mathrm{IV}] + \frac{[\mathrm{V}]-[\mathrm{VI}]-[\mathrm{VII}]+[\mathrm{VIII}]}{2}
\end{aligned}
$$

6. Lower Bound : General Case

The purpose of this section is to prove the following theorem.

Theorem 4. Any algorithm which computes the quaternion product over arbitrary rings using scalars from F requires at least eight active multiplications.

Recall the assumption (*) about F: for all $a,b,c,d,\in F$, $a^2+b^2+c^2+d^2=0$ if and only if $a=b=c=d=0$. The proof will use this assumption in several places. Two lemmas contain most of the proof.

Lemma 1. If there is an algorithm which computes the quaternion product using q active multiplications, then there is also an algorithm of the same complexity in which one active multiplication is by x_1 and another is by y_1.

Proof. Let a_1,a_2,a_3,a_4 be elements of F, not all zero. Let $\mathrm{d}(a)=a_1^2+a_2^2+a_3^2+a_4^2 \neq 0$ and

$$
A = 1/\mathrm{d}(a)\begin{bmatrix} a_1 & -a_2 & -a_3 & -a_4 \\ a_2 & a_1 & -a_4 & a_3 \\ a_3 & a_4 & a_1 & -a_2 \\ a_4 & -a_3 & a_2 & a_1 \end{bmatrix}
$$

Let I be the 4×4 identity matrix. A direct computation shows that $T_A=(A,I,\mathrm{d}(a)A)$ is a transformation on the quaternion product.

If there is an algorithm computing the quaternion product using q active multiplications, then by Theorem 1 there is also one of the form

$$W \begin{bmatrix} (u_1^t x)\cdot(v_1^t y) \\ \cdot \\ \cdot \\ \cdot \\ (u_q^t x)\cdot(v_q^t y) \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}$$

.

Call this algorithm α. Note that no u_i or v_i can be the zero vector, otherwise the corresponding multiplication would not be active.

Let $u_1^T = [u_{11} \quad u_{21} \quad u_{31} \quad u_{41}]$. Then the transformation T_A with $[a_1 \quad a_2 \quad a_3 \quad a_4] = [u_{11} \quad u_{21} \quad u_{31} \quad u_{41}]$ when applied to α produces a new algorithm α' in which the first multiplication is $u_1^t(Ax)\cdot v_1^t(Iy) = x_1 \cdot v_1^t y$.

Similarly, $T_B = (I, B, \mathrm{d}(b)B)$ is a transformation if

$$B = 1/\mathrm{d}(b) \begin{bmatrix} b_1 & -b_2 & -b_3 & -b_4 \\ b_2 & b_1 & b_4 & -b_3 \\ b_3 & -b_4 & b_1 & b_2 \\ b_4 & b_3 & -b_2 & b_1 \end{bmatrix}$$

and $b_1, b_2, b_3, b_4 \in F$ are not all zero. When $[b_1 \quad b_2 \quad b_3 \quad b_4] = v_2^t$, T_B transforms α' into α'' in which the first two multiplications are $x_1 \cdot v_1''^t y$ and $u_2''^t x \cdot y_1$, where $v_1''^t = v_1^t B$ and $u_2''^t = u_2^t A$.

Lemma 2. Any algorithm which computes the quaternion product of $x_2\mathbf{i} + x_3\mathbf{j} + x_4\mathbf{k}$ and $y_2\mathbf{i} + y_3\mathbf{j} + y_4\mathbf{k}$ in the general case, using scalars from F requires at least six active multiplications.

Proof. The trilinear form T' associated with this computation is:

$$\begin{aligned} T' = {} & (-x_2y_2 - x_3y_3 - x_4y_4)z_1 + (x_3y_4 - x_4y_3)z_2 \\ & + (-x_2y_4 + x_4y_2)z_3 + (x_2y_3 - x_3y_2)z_4 \end{aligned}$$

By Theorem 2, the least number of active multiplications needed to compute this product is equal to the tensor rank of the matrix

$$M'(z) = \begin{bmatrix} -z_1 & z_4 & -z_3 \\ -z_4 & -z_1 & z_2 \\ z_3 & -z_2 & -z_1 \end{bmatrix}$$

As the determinant of this matrix is equal to $-z_1(z_1^2 + z_2^2 + z_3^2 + z_4^2)$, $M'(z)$ is of rank three if $z_1 \neq 0$.

Let q be the tensor rank of $M'(z)$. $M'(z) = \sum_{\ell=1}^{q} u_\ell v_\ell^t (w_\ell^t z)$, $u_\ell \in F^3, v_\ell \in F^3, w_\ell \in F^4$. There must exist four linearly independent vectors among the $w_\ell, (\ell = 1, ..., q)$, as $M'(z)$ depends on four independent parameters. Assume w_1, w_2, w_3, w_4 are linearly independent. There exists z° with $z_1^\circ \neq 0$ such that:

$$w_1^t z^\circ = 0, \qquad w_2^t z^\circ = 0, \qquad w_3^t z^\circ = 0.$$

For this choice of z°,

$$M'(z^\circ) = \sum_{\ell=4}^{q} u_\ell v_\ell^t (w_\ell^t z^\circ)$$

Since $z_1^\circ \neq 0$, $M'(z^\circ)$ is of rank three and therefore $3 \leq q-3$ which implies $q \geq 6$.

Proof of Theorem 4. Suppose that there is an algorithm computing the quaternion product with only 7 active multiplications. By Lemma 1 there is another algorithm requiring only 7 in which one active multiplication is by x_1 and one is by y_1. Setting $x_1 = y_1 = 0$ in this algorithm produces a new algorithm computing the product of $x_2\mathbf{i} + x_3\mathbf{j} + x_4\mathbf{k}$ and $y_2\mathbf{i} + y_3\mathbf{j} + y_4\mathbf{k}$ with only 5 active multiplications. By Lemma 4 this is impossible.

The algorithms given in sections 4 and 5 attain the lower bound given by Theorem 4. Thus, the complexity of the quaternion product in the general case is exactly eight. Combined with Theorem 2 this fact can be restated in the following way:

Corollary. The tensor rank of

$$M(z) = \begin{bmatrix} z_1 & z_2 & z_3 & z_4 \\ z_2 & -z_1 & z_4 & -z_3 \\ z_3 & -z_4 & -z_1 & z_2 \\ z_4 & z_3 & -z_2 & -z_1 \end{bmatrix}$$

is eight.

7. Lower Bound : Commutative Case

Theorem 5. Any algorithm which computes the quaternion product over commutative rings using scalars from F requires at least seven active multiplications.

Proof. Let q be the minimum number of active multiplications required. Theorem 3 states that q is the minimum tensor rank of any $N(z)$ satisfying

$$N(z) + N(z)^t = \begin{bmatrix} 0 & M(z) \\ M(z)^t & 0 \end{bmatrix}. \tag{1}$$

($M(z)$ is defined in section 4). Then we can write

$$N(z) = \sum_{\ell=1}^{q} u_\ell v_\ell^t (w_\ell^t z), u_\ell \in F^8, v_\ell \in F^8, w_\ell \in F^4 \quad \text{for} \quad \ell = 1, ..., q. \tag{2}$$

The matrix $N(z)$ depends on four independent parameters $z_1, ..., z_4$ (from (1)). Therefore we must have four vectors w_ℓ linearly independent in the expression (2) for $N(z)$. Assume w_1, w_2, w_3 and w_4 are linearly independent. The components of z can be assigned values from F to satisfy the following system of equations:

$$w_1^t z = 0, \qquad w_2^t z = 0, \qquad w_3^t = 0.$$

Let $z^\circ \in F^4$ be a nontrivial solution of this system ($z^\circ \neq 0$):

$$z^{\circ t} = [\, z_1^\circ \quad z_2^\circ \quad z_3^\circ \quad z_4^\circ \,].$$

For this choice of z,

$$N(z^\circ) = \sum_{\ell=4}^{q} u_\ell v_\ell^t (w_\ell^t z^\circ).$$

$N(z^\circ)$ is expressed as a linear combination of $q-3$ rank-one matrices, therefore

$$\mathrm{Rank}(N(z^\circ)) \leq q-3.$$

But we have also:

$$N(z^\circ) + N(z^\circ)^t = \begin{bmatrix} 0 & M(z^\circ) \\ M(z^\circ)^t & 0 \end{bmatrix}.$$

A brief calculation shows that

$$\det(M(z^\circ)) = -((z_1^\circ)^2 + (z_2^\circ)^2 + (z_3^\circ)^2 + (z_4^\circ)^2)^2.$$

From the fact that z° is an element of F^4 not equal to zero and assumption (*), we deduce that $\det(M(z^\circ)) \neq 0$, and therefore

$$\mathrm{Rank}(N(z^\circ) + N(z^\circ)^t) = 8.$$

But $\mathrm{Rank}(N(z^\circ) + N(z^\circ)^t) \leq 2\,\mathrm{Rank}(N(z^\circ)) \leq 2(q-3)$, so $8 \leq 2(q-3)$ implies $q \geq 7$.

8. A Counterexample

The proofs of the lower bounds in both the general and commutative cases depend on the assumption (*) that the equation, $a^2+b^2+c^2+d^2=0$, has no nontrivial solutions in F. For example, (*) does not hold if F is the complex numbers. The following example will show that Theorem 4 is not true with this choice of F.

With the real quaternion $X = x_1 + x_2\mathbf{i} + x_3\mathbf{j} + x_4\mathbf{k}$ we can associate the following 2×2 complex matrix.

$$\begin{bmatrix} X_1 & X_2 \\ -\bar{X}_2 & \bar{X}_1 \end{bmatrix} \qquad X_1 = x_1 + x_2 i, \quad X_2 = x_3 + x_4 i$$

This mapping is an isomorphism between quaternions and the set of matrices of this type. The product of two quaternions can be viewed as the product of two such complex matrices. From the result of Strassen [1], this product can be performed in seven complex multiplications (in fact, six complex and one real multiplication).[3]

The same construction yields an algorithm for multiplying quaternions over an arbitrary ring with seven "complex" multiplications, where a "complex" multiplication means the construction of $(x_1y_1 - x_2y_2) + (x_1y_2 + x_2y_1)i$ from $x_1 + x_2i$ and $y_1 + y_2i$, and x_1, x_2, y_1 and y_2 are ring elements.

9. Conclusion

For the general case the complexity of the quaternion product is exactly eight. For the commutative case the complexity is either seven or eight. No algorithm is known which uses commutativity to advantage in computing the quaternion product. Such an algorithm or a proof that none exists would eliminate the gap between the upper and lower bounds in the commutative case.

It has recently come to the authors' attention that some of the results reported here have been previously obtained. The upper bound of section 4 appears in Dobkin [11]. The lower bound for the general case has been proven by de Groote [12].

References

[1] Strassen, V., Gaussian elimination is not optimal, Numer. Math., 13 (1969), 354–356.

[2] Hopcroft, J.E., and L.R. Kerr, On minimizing the number of multiplications necessary for matrix multiplication, SIAM J. Appl. Math., 20 (1971), 30–35.

[3] Winograd, S., On multiplication of 2×2 matrices, Linear Algebra and its Applications, 4 (1971), 381-388.

[4] Fiduccia, C.M., On obtaining upper bounds on the complexity of matrix multiplication, *Complexity of Computer Computations*, Plenum Press, New York, 1972, 3–40.

[5] de Groote, H.F., On the complexity of quaternion multiplication, Tägung über algorithmen und komplexitätstheorie, Oberwolfach, (November 1974).

[6] Gastinel, N., On the simultaneous calculation of a set of bilinear forms, Gatlinburg V Symposium on Numerical Algebra, (June 1972).

[7] Musinski, J., Determining the complexity of matrix multiplication and other bilinear forms, Ph.D. Thesis, Cornell University, (May 1973).

[8] Aho, A.V., J.E. Hopcroft, and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading, Mass., 1974, 446.

[3] Editor's note: This isomorphism between real quaternions and the pair X_1, X_2 of complex numbers in the first row of the matrix also shows that the real quaternion product requires at most 4 complex multiplications. The top row of the result is obtained with 4 complex multiplications: $X_1Y_1 - X_2\bar{Y}_2, X_1Y_2 + X_2\bar{Y}_1$. The bottom row of the result matrix requires no additional multiplications, because it can be obtained from the top row by trivial operations.

[9] Lafon, J.C., Optimum computation of p bilinear forms, (to appear in Journal of Linear Algebra), (1973).

[10] Strassen, V., Vermeidung von divisionen, Crelle Journal für die Reine und Angewante Mathematik, (1973).

[11] Dobkin, D., On the arithmetic complexity of a class of arithmetic computations, Ph.D. Thesis, Harvard University, 1973.

[12] de Groote, H.F., On the computational complexity of quaternion multiplication, preprint, 1975.