

# STRASSEN'S MATRIX MULTIPLICATION ALGORITHM FOR MATRICES OF ARBITRARY ORDER

IVO HEDTKE

**ABSTRACT.** The well known algorithm of VOLKER STRASSEN for matrix multiplication can only be used for  $(m2^k \times m2^k)$  matrices. For arbitrary  $(n \times n)$  matrices one has to add zero rows and columns to the given matrices to use STRASSEN's algorithm. STRASSEN gave a strategy of how to set  $m$  and  $k$  for arbitrary  $n$  to ensure  $n \leq m2^k$ . In this paper we study the number  $d$  of additional zero rows and columns and the influence on the number of flops used by the algorithm in the worst case ( $d = n/16$ ), best case ( $d = 1$ ) and in the average case ( $d \approx n/48$ ). The aim of this work is to give a detailed analysis of the number of additional zero rows and columns and the additional work caused by STRASSEN's bad parameters. STRASSEN used the parameters  $m$  and  $k$  to show that his matrix multiplication algorithm needs less than  $4.7n^{\log_2 7}$  flops. We can show in this paper, that these parameters cause an additional work of approximately 20 % in the worst case in comparison to the optimal strategy for the worst case. This is the main reason for the search for better parameters.

## 1. INTRODUCTION

In his paper “*Gaussian Elimination is not Optimal*” ([14]) VOLKER STRASSEN developed a recursive algorithm (we will call it  $\mathcal{S}$ ) for multiplication of square matrices of order  $m2^k$ . The algorithm itself is described below. Further details can be found in [8, p. 31].

Before we start with our analysis of the parameters of STRASSEN's algorithm we will have a short look on the history of fast matrix multiplication. The naive algorithm for matrix multiplication is an  $\mathcal{O}(n^3)$  algorithm. In 1969 STRASSEN showed that there is an  $\mathcal{O}(n^{2.81})$  algorithm for this problem. SHMUEL WINOGRAD optimized STRASSEN's algorithm. While the STRASSEN-WINOGRAD algorithm is a variant that is always implemented (for example in the famous GEMMW package), there are faster ones (in theory) that are impractical to implement. The fastest known algorithm, devised in 1987 by DON COPPERSMITH and WINOGRAD, runs in  $\mathcal{O}(n^{2.38})$  time. There is also an interesting group-theoretic approach to fast matrix multiplication from HENRY COHN and CHRISTOPHER UMANS, see [4], [5] and [13]. Most researchers believe that an optimal algorithm with  $\mathcal{O}(n^2)$  runtime exists, since then no further progress was made in finding one.

Because modern architectures have complex memory hierarchies and increasing parallelism, performance has become a complex tradeoff, not just a simple matter of counting flops (in this article one flop means one floating-point operation, that means one addition is a flop and one multiplication is one flop, too). Algorithms

---

2010 *Mathematics Subject Classification.* Primary 65F30; Secondary 68Q17.

*Key words and phrases.* Fast Matrix Multiplication, Strassen Algorithm.

The author was supported by the Studienstiftung des Deutschen Volkes.

which make use of this technology were described by PAOLO D'ALBERTO and ALEXANDRU NICOLAU in [1]. An also well known method is *Tiling*: The normal algorithm can be speeded up by a factor of two by using a six loop implementation that blocks submatrices so that the data passes through the L1 Cache only once.

**1.1. The algorithm.** Let  $A$  and  $B$  be  $(m2^k \times m2^k)$  matrices. To compute  $C := AB$  let

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},$$

where  $A_{ij}$  and  $B_{ij}$  are matrices of order  $m2^{k-1}$ . With the following auxiliary matrices

$$\begin{aligned} H_1 &:= (A_{11} + A_{22})(B_{11} + B_{22}) & H_2 &:= (A_{21} + A_{22})B_{11} \\ H_3 &:= A_{11}(B_{12} - B_{22}) & H_4 &:= A_{22}(B_{21} - B_{11}) \\ H_5 &:= (A_{11} + A_{12})B_{22} & H_6 &:= (A_{21} - A_{11})(B_{11} + B_{12}) \\ H_7 &:= (A_{12} - A_{22})(B_{21} + B_{22}) \end{aligned}$$

we get

$$C = \begin{bmatrix} H_1 + H_4 - H_5 + H_7 & H_3 + H_5 \\ H_2 + H_4 & H_1 + H_3 - H_2 + H_6 \end{bmatrix}.$$

This leads to recursive computation. In the last step of the recursion the products of the  $(m \times m)$  matrices are computed with the naive algorithm (straight forward implementation with three `for`-loops, we will call it  $\mathcal{N}$ ).

**1.2. Properties of the algorithm.** The algorithm  $\mathcal{S}$  needs (see [14])

$$F_S(m, k) := 7^k m^2 (2m + 5) - 4^k 6m^2$$

flops to compute the product of two square matrices of order  $m2^k$ . The naive algorithm  $\mathcal{N}$  needs  $n^3$  multiplications and  $n^3 - n^2$  additions to compute the product of two  $(n \times n)$  matrices. Therefore  $F_N(n) = 2n^3 - n^2$ . In the case  $n = 2^p$  the algorithm  $\mathcal{S}$  is better than  $\mathcal{N}$  if  $F_S(1, p) < F_N(2^p)$ , which is the case iff  $p \geq 10$ . But if we use algorithm  $\mathcal{S}$  only for matrices of order at least  $2^{10} = 1024$ , we get a new problem:

**Lemma.** *The algorithm  $\mathcal{S}$  needs  $\frac{17}{3}(7^p - 4^p)$  units of memory (we write “uom” in short) (number of floats or doubles) to compute the product of two  $(2^p \times 2^p)$  matrices.*

*Proof.* Let  $M(n)$  be the number of uom used by  $\mathcal{S}$  to compute the product of matrices of order  $n$ . The matrices  $A_{ij}$ ,  $B_{ij}$  and  $H_\ell$  need  $15(n/2)^2$  uom. During the computation of the auxiliary matrices  $H_\ell$  we need  $7M(n/2)$  uom and  $2(n/2)^2$  uom as input arguments for the recursive calls of  $\mathcal{S}$ . Therefore we get  $M(n) = 7M(n/2) + (17/4)n^2$ . Together with  $M(1) = 0$  this yields to  $M(2^p) = \frac{17}{3}(7^p - 4^p)$ .  $\square$

As an example, if we compute the product of two  $(2^{10} \times 2^{10})$  matrices (represented as `double` arrays) with  $\mathcal{S}$  we need  $8 \cdot \frac{17}{3}(7^{10} - 4^{10})$  bytes, i.e. 12.76 gigabytes of memory. That is an enormous amount of RAM for such a problem instance. BRICE BOYER et al. ([3]) solved this problem with fully in-place schedules of STRASSEN-WINOGRADE's algorithm (see the following paragraph), if the input matrices can be overwritten.

SHMUEL WINOGRAD optimized STRASSEN's algorithm. The STRASSEN-WINOGRAD algorithm (described in [11]) needs only 15 additions and subtractions, whereas  $\mathcal{S}$  needs 18. WINOGRAD had also shown (see [15]), that the minimum number of multiplications required to multiply  $2 \times 2$  matrices is 7. Furthermore, ROBERT PROBERT ([12]) showed that 15 additive operations are necessary and sufficient to multiply two  $2 \times 2$  matrices with 7 multiplications.

Because of the bad properties of  $\mathcal{S}$  with full recursion and large matrices, one can study the idea to use only one step of recursion. If  $n$  is even and we use one step of recursion of  $\mathcal{S}$  (for the remaining products we use  $\mathcal{N}$ ) the ratio of this operation count to that required by  $\mathcal{N}$  is (see [9])

$$\frac{7n^3 + 11n^2}{8n^3 - 4n^2} \xrightarrow{n \rightarrow \infty} \frac{7}{8}.$$

Therefore the multiplication of two sufficiently large matrices using  $\mathcal{S}$  costs approximately 12.5 % less than using  $\mathcal{N}$ .

Using the technique of stopping the recursion in the STRASSEN-WINOGRAD algorithm early, there are well known implementations, as for example

- on the Cray-2 from DAVID BAILEY ([2]),
- GEMMW from DOUGLAS et al. ([7]) and
- a routine in the IBM ESSL library routine ([10]).

**1.3. The aim of this work.** STRASSEN's algorithm can only be used for  $(m2^k \times m2^k)$  matrices. For arbitrary  $(n \times n)$  matrices one has to add zero rows and columns to the given matrices (see the next section) to use STRASSEN's algorithm. STRASSEN gave a strategy of how to set  $m$  and  $k$  for arbitrary  $n$  to ensure  $n \leq m2^k$ . In this paper we study the number  $d$  of additional zero rows and columns and the influence on the number of flops used by the algorithm in the worst case, best case and in the average case.

It is known ([11]), that these parameters are not optimal. We only study the number  $d$  and the additional work caused by the bad parameters of STRASSEN. We give no better strategy of how to set  $m$  and  $k$ , and we do not analyze other strategies than the one from STRASSEN.

## 2. STRASSEN'S PARAMETER FOR MATRICES OF ARBITRARY ORDER

Algorithm  $\mathcal{S}$  uses recursions to multiply matrices of order  $m2^k$ . If  $k = 0$  then  $\mathcal{S}$  coincides with the naive algorithm  $\mathcal{N}$ . So we will only consider the case where  $k > 0$ . To use  $\mathcal{S}$  for arbitrary  $(n \times n)$  matrices  $A$  and  $B$  (that means for arbitrary  $n$ ) we have to embed them into matrices  $\tilde{A}$  and  $\tilde{B}$  which are both  $(\tilde{n} \times \tilde{n})$  matrices with  $\tilde{n} := m2^k \geq n$ . We do this by adding  $\ell := \tilde{n} - n$  zero rows and columns to  $A$  and  $B$ . This results in

$$\tilde{A}\tilde{B} = \begin{bmatrix} A & 0^{n \times \ell} \\ 0^{\ell \times n} & 0^{\ell \times \ell} \end{bmatrix} \begin{bmatrix} B & 0^{n \times \ell} \\ 0^{\ell \times n} & 0^{\ell \times \ell} \end{bmatrix} =: \tilde{C},$$

where  $0^{k \times j}$  denotes the  $(k \times j)$  zero matrix. If we delete the last  $\ell$  columns and rows of  $\tilde{C}$  we get the result  $C = AB$ .

We now focus on how to find  $m$  and  $k$  for arbitrary  $n$  with  $n \leq m2^k$ . An optimal but purely theoretical choice is

$$(m^*, k^*) = \arg \min \{F_{\mathcal{S}}(m, k) : (m, k) \in \mathbb{N} \times \mathbb{N}_0, n \leq m2^k\}.$$

Further methods of finding  $m$  and  $k$  can be found in [11]. We choose another way. According to STRASSEN's proof of the main result of [14], we define

$$k := \lfloor \log_2 n \rfloor - 4 \quad \text{and} \quad m := \lfloor n2^{-k} \rfloor + 1, \quad (2.1)$$

where  $\lfloor x \rfloor$  denotes the largest integer not greater than  $x$ . We define  $\tilde{n} := m2^k$  and study the relationship between  $n$  and  $\tilde{n}$ . The results are:

- *worst case*:  $\tilde{n} \leq (17/16)n$ ,
- *best case*:  $\tilde{n} \geq n + 1$  and
- *average case*:  $\tilde{n} \approx (49/48)n$ .

### 2.1. Worst case analysis.

**Theorem 2.1.** *Let  $n \in \mathbb{N}$  with  $n \geq 16$ . For the parameters (2.1) and  $m2^k = \tilde{n}$  we have*

$$\tilde{n} \leq \frac{17}{16}n.$$

*If  $n$  is a power of two, we have  $\tilde{n} = \frac{17}{16}n$ .*

*Proof.* For fixed  $n$  there is exactly one  $\alpha \in \mathbb{N}$  with  $2^\alpha \leq n < 2^{\alpha+1}$ . We define  $I^\alpha := \{2^\alpha, \dots, 2^{\alpha+1} - 1\}$ . Because of (2.1) for each  $n \in I^\alpha$  the value of  $k$  is

$$k = \lfloor \log_2 n \rfloor - 4 = \log_2 2^\alpha - 4 = \alpha - 4.$$

Possible values for  $m$  are

$$m = \left\lfloor n \frac{1}{2^{\alpha-4}} \right\rfloor + 1 = \left\lfloor n \frac{16}{2^\alpha} \right\rfloor + 1 =: m(n).$$

$m(n)$  is increasing in  $n$  and  $m(2^\alpha) = 17$  and  $m(2^{\alpha+1}) = 33$ . Therefore we have  $m \in \{17, \dots, 32\}$ . For each  $n \in I^\alpha$  one of the following inequalities holds:

$$\begin{array}{lll} (I_1^\alpha) & 2^\alpha = 16 \cdot 2^{\alpha-4} & \leq n < 17 \cdot 2^{\alpha-4} \\ (I_2^\alpha) & 17 \cdot 2^{\alpha-4} & \leq n < 18 \cdot 2^{\alpha-4} \\ & \vdots & \\ (I_{16}^\alpha) & 31 \cdot 2^{\alpha-4} & \leq n < 32 \cdot 2^{\alpha-4} = 2^{\alpha+1}. \end{array}$$

Note that  $I^\alpha = \biguplus_{j=1}^{16} I_j^\alpha$ . It follows, that for all  $n \in \mathbb{N}$  there exists exactly one  $\alpha$  with  $n \in I^\alpha$  and for all  $n \in I^\alpha$  there is exactly one  $j$  with  $n \in I_j^\alpha$ .

Note that for all  $n \in I_j^\alpha$  we have  $k = \alpha - 4$  and  $m(n) = j + 16$ . If we only focus on  $I_j^\alpha$  the difference  $\tilde{n} - n$  has its maximum at the lower end of  $I_j^\alpha$  ( $\tilde{n}$  is constant and  $n$  has its minimum at the lower end of  $I_j^\alpha$ ). On  $I_j^\alpha$  the value of  $\tilde{n}$  and the minimum of  $n$  are

$$\tilde{n}_j^\alpha := (16 + j) \cdot 2^{\alpha-4} \quad \text{and} \quad n_j^\alpha := (15 + j) \cdot 2^{\alpha-4}.$$

Therefore the difference  $d_j^\alpha := \tilde{n}_j^\alpha - n_j^\alpha$  is constant:

$$d_j^\alpha = (16 + j) \cdot 2^{\alpha-4} - (15 + j) \cdot 2^{\alpha-4} = 2^{\alpha-4} \quad \text{for all } j.$$

To set this in relation with  $n$  we study

$$r_j^\alpha := \frac{\tilde{n}_j^\alpha}{n_j^\alpha} = \frac{n_j^\alpha + d_j^\alpha}{n_j^\alpha} = 1 + \frac{d_j^\alpha}{n_j^\alpha} = 1 + \frac{2^{\alpha-4}}{n_j^\alpha}.$$

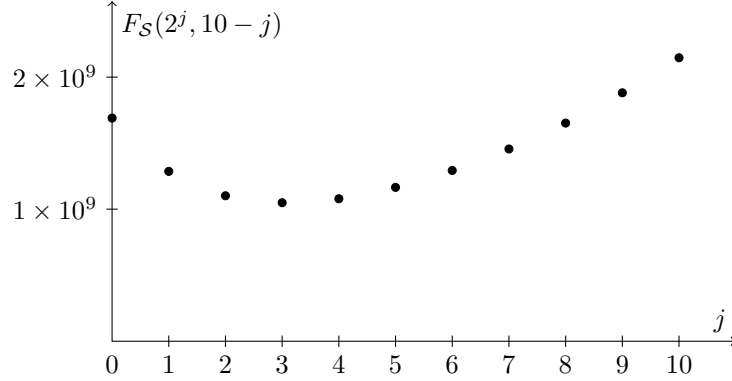


FIGURE 1. Different parameters ( $m = 2^j$ ,  $k = 10 - j$ ,  $n = m2^k$ ) to apply in the STRASSEN algorithm for matrices of order  $2^{10}$ .

Finally  $r_j^\alpha$  is maximal, iff  $n_j^\alpha$  is minimal, which is the case for  $n_1^\alpha = 16 \cdot 2^{\alpha-4} = 2^\alpha$ . With  $r_1^\alpha = 17/16$  we get  $\tilde{n} \leq \frac{17}{16}n$ , which completes the proof.  $\square$

Now we want to use the result above to take a look at the number of flops we need for  $\mathcal{S}$  in the worst case. The worst case is  $n = 2^p$  for any  $4 \leq p \in \mathbb{N}$ . An optimal decomposition (in the sense of minimizing the number  $(\tilde{n} - n)$  of zero rows and columns we add to the given matrices) is  $m = 2^j$  and  $k = p - j$ , because  $m2^k = 2^j 2^{p-j} = 2^p = n$ . Note that these parameters  $m$  and  $k$  have nothing to do with equation (2.1). Lets have a look on the influence of  $j$ :

**Lemma.** *Let  $n = 2^p$ . In the decomposition  $n = m2^k$  we use  $m = 2^j$  and  $k = p - j$ . Then  $f(j) := F_S(2^j, p - j)$  has its minimum at  $j = 3$ .*

*Proof.* We have  $f(j) = 2 \cdot 7^p (8/7)^j + 5 \cdot 7^p (4/7)^j - 4^p 6$ . Thus

$$\begin{aligned} f(j+1) - f(j) &= [2 \cdot 7^p (8/7)^{j+1} + 5 \cdot 7^p (4/7)^{j+1} - 4^p 6] - [2 \cdot 7^p (8/7)^j + 5 \cdot 7^p (4/7)^j - 4^p 6] \\ &= 2 \cdot 7^p (8/7)^j (8/7 - 1) + 5 \cdot 7^p (4/7)^j (4/7 - 1) \\ &= 2 \cdot 7^p (8/7)^j \cdot 1/7 - 15 \cdot 7^p (4/7)^j \cdot 1/7 = 2(4/7)^j 7^{p-1} (2^j - 7.5). \end{aligned}$$

Therefore,  $f(j)$  is a minimum if  $j = \min\{i : 2^i - 7.5 > 0\} = 3$ .  $\square$

Figure 1 shows that it is not optimal to use  $\mathcal{S}$  with full recursion in the example  $p = 10$ . Now we study the worst case  $n = 2^p$  and different sets of parameters  $m$  and  $k$  for  $F_S(m, k)$ :

- (1) If we use equation (2.1), we get the original parameters of STRASSEN:  $k = p - 4$  and  $m = 17$ . Therefore we define

$$F_1(p) := F_S(17, p - 4) = 7^p \frac{39 \cdot 17^2}{7^4} - 4^p \frac{6 \cdot 17^2}{4^4}.$$

- (2) Obviously  $m = 16$  would be a better choice, because with this we get  $m2^k = n$  (we avoid the additional zero rows and columns). Now we define

$$F_2(p) := F_S(16, p - 4) = 7^p \frac{37 \cdot 16^2}{7^4} - 4^p 6.$$

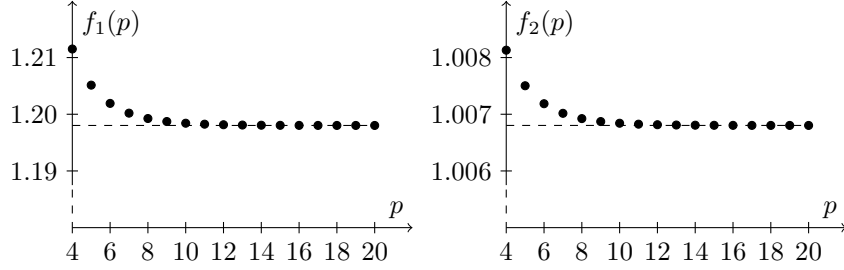


FIGURE 2. Comparison of STRASSEN's parameters ( $f_1$ ,  $m = 17$ ,  $k = p - 4$ ), obviously better parameters ( $f_2$ ,  $m = 16$ ,  $k = p - 4$ ) and the optimal parameters (lemma,  $m = 8$ ,  $k = p - 3$ ) for the worst case  $n = 2^p$ . Limits of  $f_j$  are dashed.

(3) Finally we use the lemma above. With  $m = 8 = 2^3$  and  $k = p - 3$  we get

$$F_3(p) := F_S(8, p - 3) = 7^p \frac{3 \cdot 64}{49} - 4^p 6.$$

Now we analyze  $F_i$  relative to each other. Therefore we define  $f_j := F_j/F_3$  ( $j = 1, 2$ ). So we have  $f_j: \{4, 5, \dots\} \rightarrow \mathbb{R}$ , which is monotonously decreasing in  $p$ . With

$$f_1(p) = \frac{289(4^p \cdot 2401 - 7^p \cdot 1664)}{12544(4^p \cdot 49 - 7^p \cdot 32)} \quad \text{and} \quad f_2(p) = \frac{4^p \cdot 7203 - 7^p \cdot 4736}{147(4^p \cdot 49 - 7^p \cdot 32)}$$

we get

$$\begin{aligned} f_1(4) &= 3179/2624 \approx 1.2115 & \lim_{p \rightarrow \infty} f_1(p) &= 3757/3136 \approx 1.1980 \\ f_2(4) &= 124/123 \approx 1.00813 & \lim_{p \rightarrow \infty} f_2(p) &= 148/147 \approx 1.00680. \end{aligned}$$

Figure 2 shows the graphs of the functions  $f_j$ . In conclusion, in the worst case the parameters of STRASSEN need approx. 20 % more flops than the optimal parameters of the lemma.

## 2.2. Best case analysis.

**Theorem 2.2.** *Let  $n \in \mathbb{N}$  with  $n \geq 16$ . For the parameters (2.1) and  $m2^k = \tilde{n}$  we have*

$$n + 1 \leq \tilde{n}.$$

*If  $n = 2^p \ell - 1$ ,  $\ell \in \{16, \dots, 31\}$ , we have  $\tilde{n} = n + 1$ .*

*Proof.* Like in Theorem 2.1 we have for each  $I_j^\alpha$  a constant value for  $\tilde{n}_j^\alpha$  namely  $2^{\alpha-4}(16+j)$ . Therefore  $n < \tilde{n}$  holds. The difference  $\tilde{n} - n$  has its minimum at the upper end of  $I_j^\alpha$ . There we have  $\tilde{n} - n = 2^{\alpha-4}(16+j) - (2^{\alpha-4}(16+j) - 1) = 1$ . This shows  $n + 1 \leq \tilde{n}$ .  $\square$

Let us focus on the flops we need for  $\mathcal{S}$ , again. Lets have a look at the example  $n = 2^p - 1$ . The original parameters (see equation (2.1)) for  $\mathcal{S}$  are  $k = p - 5$  and  $m = 32$ . Accordingly we define  $F(p) := F_S(32, p - 5)$ . Because  $2^p - 1 \leq 2^p$  we can add 1 zero row and column and use the lemma from the worst case. Now we get

the parameters  $m = 8$  and  $k = p - 3$  and define  $\tilde{F}(p) := F_S(8, p - 3)$ . To analyze  $F$  and  $\tilde{F}$  relative to each other we have

$$r(p) := \frac{F(p)}{\tilde{F}(p)} = \frac{4^p \cdot 16807 - 7^p \cdot 11776}{4^p \cdot 16807 - 7^p \cdot 10976}.$$

Note that  $r: \{5, 6, \dots\} \rightarrow \mathbb{R}$  is monotonously decreasing in  $p$  and has its maximum at  $p = 5$ . We get

$$r(5) = 336/311 \approx 1.08039 \quad \lim_{p \rightarrow \infty} r(p) = 11776/10976 \approx 1.07289.$$

Therefore we can say: In the best case the parameters of STRASSEN are approx. 8 % worse than the optimal parameters from the lemma in the worst case.

**2.3. Average case analysis.** With  $\mathbb{E}\tilde{n}$  we denote the expected value of  $\tilde{n}$ . We search for a relationship like  $\tilde{n} \approx \gamma n$  for  $\gamma \in \mathbb{R}$ . That means  $\mathbb{E}[\tilde{n}/n] = \gamma$ .

**Theorem 2.3.** *For the parameters (2.1) of STRASSEN  $m2^k = \tilde{n}$  we have*

$$\mathbb{E}\tilde{n} = \frac{49}{48}n.$$

*Proof.* First we focus only on  $I^\alpha$ . We write  $\mathbb{E}^\alpha := \mathbb{E}|_{I^\alpha}$  and  $\mathbb{E}_j^\alpha := \mathbb{E}|_{I_j^\alpha}$  for the expected value on  $I^\alpha$  and  $I_j^\alpha$ , resp. We have

$$\mathbb{E}^\alpha \tilde{n} = \frac{1}{16} \sum_{j=1}^{16} \mathbb{E}_j^\alpha \tilde{n} = \frac{1}{16} \sum_{j=1}^{16} \tilde{n}_j^\alpha = \frac{1}{16} \sum_{j=1}^{16} (j+16)2^{\alpha-4} = 2^{\alpha-5} \cdot 49.$$

Together with  $\mathbb{E}^\alpha n = \frac{1}{2}[(2^{\alpha+1} - 1) + 2^\alpha] = 2^\alpha + 2^{\alpha-1} - 1/2$ , we get

$$\rho(\alpha) := \mathbb{E}^\alpha \left[ \frac{\tilde{n}}{n} \right] = \frac{\mathbb{E}^\alpha \tilde{n}}{\mathbb{E}^\alpha n} = \frac{2^{\alpha-5} \cdot 49}{2^\alpha + 2^{\alpha-1} - 1/2}.$$

Now we want to calculate  $\mathbb{E}_k := \mathbb{E}|_{U(k)}[\tilde{n}/n]$ , where  $U(k) := \biguplus_{j=0}^k I^{4+j}$  by using the values  $\rho(j)$ . Because of  $|I^5| = 2|I^4|$  and  $|I^4 \cup I^5| = 3|I^4|$  we have  $\mathbb{E}_1 = \frac{1}{3}\rho(4) + \frac{2}{3}\rho(5)$ . With the same argument we get

$$\mathbb{E}_k = \sum_{j=4}^{4+k} \beta_j \rho(j) \quad \text{where} \quad \beta_j = \frac{2^{j-4}}{2^{k+1} - 1}.$$

Finally we have

$$\begin{aligned} \mathbb{E} \left[ \frac{\tilde{n}}{n} \right] &= \lim_{k \rightarrow \infty} \mathbb{E}_k = \lim_{k \rightarrow \infty} \left( \sum_{j=4}^{4+k} \beta_j \rho(j) \right) \\ &= \lim_{k \rightarrow \infty} \left( \frac{49}{2^{k+1} - 1} \sum_{j=4}^{4+k} \frac{2^{2j-9}}{2^j + 2^{j-1} - 1/2} \right) = \frac{49}{48}, \end{aligned}$$

what we intended to show.  $\square$

Compared to the worst case ( $\tilde{n} \leq \frac{17}{16}n$ ,  $17/16 = 1 + 1/16$ ), note that  $49/48 = 1 + 1/48 = 1 + \frac{1}{3} \cdot \frac{1}{16}$ .

## 3. CONCLUSION

STRASSEN used the parameters  $m$  and  $k$  in the form (2.1) to show that his matrix multiplication algorithm needs less than  $4.7n^{\log_2 7}$  flops. We could show in this paper, that these parameters cause an additional work of approx. 20 % in the worst case in comparison to the optimal strategy for the worst case. This is the main reason for the search for better parameters, like in [11].

## REFERENCES

- [1] P. D'Alberto and A. Nicolau, *Adaptive Winograd's Matrix Multiplications*, ACM Trans. Math. Softw. 36, 1, Article 3 (March 2009).
- [2] D. H. Bailey, *Extra High Speed Matrix Multiplication on the Cray-2*, SIAM J. Sci. Stat. Comput., Vol. 9, Co. 3, 603–607, 1988.
- [3] B. Boyer, J.-G. Dumas, C. Pernet, and W. Zhou, *Memory efficient scheduling of Strassen-Winograd's matrix multiplication algorithm*, International Symposium on Symbolic and Algebraic Computation 2009, Séoul.
- [4] H. Cohn and C. Umans, *A Group-theoretic Approach to Fast Matrix Multiplication*, Proceedings of the 44<sup>th</sup> Annual Symposium on Foundations of Computer Science, 11-14 October 2003, Cambridge, MA, IEEE Computer Society, pp. 438-449.
- [5] H. Cohn, R. Kleinberg, B. Szegedy and C. Umans, *Group-theoretic Algorithms for Matrix Multiplication*, Proceedings of the 46<sup>th</sup> Annual Symposium on Foundations of Computer Science, 23-25 October 2005, Pittsburgh, PA, IEEE Computer Society, pp. 379-388.
- [6] D. Coppersmith and S. Winograd, *Matrix Multiplication via Arithmetic Progressions*, STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of Computing, 1987.
- [7] C. C. Douglas, M. Heroux, G. Slisman and R. M. Smith, *GEMMW: A Portable Level 3 BLAS Winograd Variant of Strassen's Matrix-Matrix Multiply Algorithm*, J. of Comp. Physics, 110:1–10, 1994.
- [8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Third ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
- [9] S. Huss-Lederman, E. M. Jacobson, J. R. Johnson, A. Tsao, and T. Turnbull, *Implementation of Strassen's Algorithm for Matrix Multiplication*, 0-89791-854-1, 1996 IEEE
- [10] *IBM Engineering and Scientific Subroutine Library Guide and Reference*, 1992. Order No. SC23-0526.
- [11] P. C. Fischer and R. L. Probert, *Efficient Procedures for using Matrix Algorithms*, Automata, Languages and Programming – 2nd Colloquium, University of Saarbrücken, Lecture Notes in Computer Science, 1974.
- [12] R. L. Probert, *On the additive complexity of matrix multiplication*, SIAM J. Compu, 5:187–203, 1976.
- [13] S. Robinson, *Toward an Optimal Algorithm for Matrix Multiplication*, SIAM News, Volume 38, Number 9, November 2005.
- [14] V. Strassen, *Gaussian Elimination is not Optimal*, Numer. Math., 13:354–356, 1969.
- [15] S. Winograd, *On Multiplication of  $2 \times 2$  Matrices*, Linear Algebra and its Applications, 4:381-388, 1971.

MATHEMATICAL INSTITUTE, UNIVERSITY OF JENA, D-07737 JENA, GERMANY  
*E-mail address:* Ivo.Hedtke@uni-jena.de