Complexity of Bilinear Problems

Markus Bläser, Saarland University

Scribe: Fabian Bendun
Editor: Markus Bläser

October 7, 2009

Contents

Introduction

Given two $n \times n$-matrices $x = (x_{ik})$ and $y = (y_{kj})$ whose entries are indeterminates over some field K, we want to compute their product $xy = (z_{ij})$. The entries z_{ij} are given be the following well-known bilinear forms:

$$z_{ij} = \sum_{k=1}^{n} x_{ik} y_{kj}; \qquad 1 \leq i, j \leq n \tag{1}$$

Each z_{ij} is the sum of n products. Thus every z_{ij} can be computed with n multiplications and $n-1$ additions. This gives an algorithm that altogether uses n^3 multiplications and $n^2(n-1)$ additions. This algorithms looks so natural and intuitive that it is very hard to imagine that there is better way to multiply matrices. In 1969, however, Strassen [Str69] found a way to multiply 2×2 Matrices with only 7 multiplications but 18 additions.

Let z_{ij}, $1 \leq i, j \leq 2$, be given by

$$\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}.$$

We compute the seven products

$$\begin{aligned} p_1 &= (x_{11} + x_{22})(y_{11} + y_{22}) \\ p_2 &= (x_{11} + x_{22})y_{11} \\ p_3 &= x_{11}(y_{12} - y_{22}) \\ p_4 &= x_{22}(-y_{11} + y_{12}) \\ p_5 &= (x_{11} + x_{12})y_{22} \\ p_6 &= (-x_{11} + x_{21})(y_{11} + y_{12}) \\ p_7 &= (x_{12} - x_{22})(y_{21} + y_{22}) \end{aligned}$$

We can express each of the z_{ij} as a linear combination of these seven products, namely,

$$\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} = \begin{pmatrix} p_1 + p_4 - p_5 + p_7 & p_3 + p_5 \\ p_2 + p_4 & p_1 + p_3 - p_2 + p_6 \end{pmatrix}$$

The number of multiplications in this algorithm is optimal (we will see this later), but already for 3×3-matrices, the optimal number of multiplication is not known. We know that it is in the interval $[19; 23]$, cf. [Blä03, Lad76].

But is it really interesting to save one multiplication but have an additional 14 additions instead?[1] The important point is that Strassen's algorithm does not only work over fields

[1] There is a variant of Strassen's algorithm that uses only 15 additions [Win71]. And one can even show that this is optimal, i.e., every algorithms that uses only seven multiplications needs at least 15 additions [Bsh95].

but also over noncommutative rings. In particular, the entries of the 2×2-matrices can we matrices itself and we can apply the algorithm recursively. And for matrices, multiplications—at least if we use the naive method—are much more expensive than additions, namely $O(n^3)$ compared to n^2.

Proposition 0.1. *One can multiply $n \times n$-matrices with $O(n^{log_2 7})$ arithmetical operations (and even without using divisions).*[2]

Proof. W.l.o.g. $n = 2^\ell, \ell \in \mathbb{N}$. If this is not the case, then we can embed our matrices into matrices whose size is the next largest power of two and fill the remaining positions with zeros. Since the algorithm does not use any divisions, subsituting an indeterminate by a concrete value will not cause a division by zero.

We will show by induction in ℓ that we can multiply with 7^ℓ multiplications and $6 \cdot (7^\ell - 4^\ell)$ additions/subtractions.

Induction start ($\ell = 1$): See above.

Induction step ($\ell - 1 \to \ell$): We think of our matrices as 2×2-matrices whose entries are $2^{\ell-1} \times 2^{\ell-1}$ matrices, i.e., we have the following block structure:

$$\left(\begin{array}{cc} \cdot & \cdot \\ \cdot & \cdot \end{array}\right) \cdot \left(\begin{array}{cc} \cdot & \cdot \\ \cdot & \cdot \end{array}\right) = \left(\begin{array}{cc} \cdot & \cdot \\ \cdot & \cdot \end{array}\right).$$

We can multiply these matrices using Strassen's algorithm with seven multiplications of $2^{\ell-1} \times 2^{\ell-1}$-matrices and 18 additions of $2^{\ell-1} \times 2^{\ell-1}$-matrices.

For the seven multiplications of the $2^{\ell-1} \times 2^{\ell-1}$-matrices, we need $7 \cdot 7^{\ell-1} = 7^\ell$ multiplications by the induction hypothesis. And we need $7 \cdot 6 \cdot (7^{\ell-1} - 4^{\ell-1})$ additions/subtractions for the seven multiplications. The 18 additions of $2^{\ell-1} \times 2^{\ell-1}$-matrices need $18 \cdot (2^{\ell-1})^2$ additions. Thus the total number of additions/subtractions is

$$7 \cdot 6 \cdot (7^{\ell-1} - 4^{\ell-1}) + 18 \cdot (2^{\ell-1})^2 = 6 \cdot (7^\ell - 7 \cdot 4^{\ell-1} + 3 \cdot 4^{\ell-1}) = 6 \cdot (7^\ell - 4^\ell).$$

This finishes the induction step. Since $7^\ell = n^{\log_2 t}$, we are done. □

[2] What is an arithmetical operation? We will make this precise in the next chapter. For the moment, we compute in the field of rational functions $K(x_{ij}, y_{ij} \mid 1 \leq i, j \leq n)$. We start with the constants from K and the indeterminates x_{ij} and y_{ij}. Then we can take any two of the elements that we computed so far and compute their product, their quotient (if the second element is not zero), their sum, or their difference. We are done if we have computed all the z_{ij} in (1).

Chapter 1

Computations and costs

1.1 Karatsuba's algorithm

Let us start with a very simple computational problem, the multiplication of univariate polynomials of degree one. We are given two polynomials $a_0 + a_1 X$ and $b_0 + b_1 X$ and we want to compute the coefficients of their product, which are given by

$$(a_0 + a_1 \cdot X) \cdot (b_0 + b_1 \cdot X) = \underbrace{a_0 b_0}_{c_0} + \underbrace{(a_0 b_1 + a_1 b_0)}_{c_1} \cdot X + \underbrace{a_1 b_1}_{c_2} \cdot X^2.$$

We here consider the coefficients of the two polynomials to be indeterminates over some field K. The coefficients of the product are rational functions (in fact, bilinear forms) in a_0, a_1, b_0, b_1, so the following model of computation seems to fit well. We have a sequence $(w_1, w_2, \ldots, w_\ell)$ of rational functions such that each w_i is either a_0, a_1, b_0, or b_1 (inputs) or a constant from K or can be expressed a $w_i = w_j \,\mathrm{op}\, w_k$ for indices $j, k < i$ and op is one of the arithmetic operations $\cdot$, $/$, $+$, or $-$.

Here is one possible computation that computes the three coefficients c_0, c_1, and c_2.

$$\begin{array}{rcl} w_1 &=& a_0 \\ w_2 &=& a_1 \\ w_3 &=& b_0 \\ w_4 &=& b_1 \\ c_0 \overset{Def}{=} w_5 &=& w_1 \cdot w_3 \\ c_2 \overset{Def}{=} w_6 &=& w_2 \cdot w_4 \\ w_7 &=& w_1 + w_2 \\ w_8 &=& w_3 + w_4 \\ w_9 &=& w_7 \cdot w_8 \\ w_{10} &=& w_5 + w_6 \\ c_1 \overset{Def}{=} w_{11} &=& w_9 - w_{10} \end{array}$$

The above computation only uses three multiplications instead of four, which the naive algorithm needs. This is also called Karatsuba's algorithm. Like Strassen's algorithm, it can be generalized to higher degree polynomials. If we have two polynomials $A(X) = \sum_{i=0}^{n} a_i X^i$ and $B(X) = \sum_{j=0}^{n} b_j X^j$ with $n = 2^\ell - 1$, then we split the two polynomials into halves, that is, $A(X) = A_0(X) + X^{(n+1)/2} A_1(X)$ with $A_0(X) = \sum_{i=0}^{(n+1)/2-1} a_i X^i$ and $A_1(X) =$

$\sum_{i=0}^{(n+1)/2-1} a_{(n+1)/2+i} X^i$ and the same for B. Then we multiply these polynomials using the above scheme with A_0 taking the role of a_0 and A_1 taking the role of a_1 and the same for B. All multiplications of polynomials of degree $(n+1)/2-1$ are then performed recursively. Let $N(n)$ denote the number of arithmetic operations that the above algorithm needs to multiply polynomial of degree $\leq n$. The algorithm above gives the following recursive equation

$$N(n) = 3 \cdot N((n+1)/2 - 1) + O(n) \qquad \text{and} \quad N(2) = 7.$$

This yields $N(n) = O(n^{\log_2 3})$. Karatsuba's algorithm again trades one multiplication for a bunch of additional additions which is bad for degree one polynomials but good in general, since polynomial addition only needs n operations but polynomial multiplication—at least when using the naive method—is much more expensive, namely, $O(n^2)$.

1.2 A general model

We provide a framework to define computations and costs that is general enough to cover all the examples that we will look at. For a set S, let $\text{fin}(S)$ denote the set of all finite subsets of S.

Definition 1.1 (Computation structure)**.** *A computation structure is a set M together with a mapping $\gamma : M \times \text{fin}(M) \to [0; \infty]$ such that*

1. $\text{im}(\gamma)$ *is well ordered, that is, every subset of* $\text{im}(\gamma)$ *has a minimum,*
2. $\gamma(w, U) = 0$, *if $w \in U$,*
3. $U \subseteq V \Rightarrow \gamma(w, V) \leq \gamma(w, U)$ *for all $w \in M$, $U, V \subseteq \text{fin}(M)$.*

M is the set of objects that we are computing with. $\gamma(w, U)$ is the cost of computing w from U "in one step". In the example of the previous section, M is the the set of all rational functions in a_0, a_1, b_0, b_1. If we want to count the number of arithmetic operations of Karatsuba's algorithm, then $\gamma(w, U) = 0$ if $w \in U$. ("There are no costs if we already computed w"). We have $\gamma(w, U) = 1$ if there are $u, v \in U$ such that $w = u \,\text{op}\, v$. ("w can be computed from u and v with one arithmetical operation.") In all other cases $\gamma(w, U) = \infty$. ("w cannot be computed "in one step" from U.)

Often, we have a set M together with some operations $\phi : M^s \to M$ of some arity s. If we assign to each such operation a cost, then this induces a computation structure in a very natural way.

Definition 1.2. *A structure $(M, \phi_1, \phi_2, ...)$ with (partial) operations $\phi_j : M^{s_j} \to M$ and a cost function $\text{¢} : \{\phi_1, \phi_2, ...\} \to [0; \infty]$ such that* $\text{im}(\text{¢})$ *is well ordered induces a computation structure in the following way:*

$$\gamma(w, U) \stackrel{Def}{=} \min\{\text{¢}(\phi_j) \mid \exists u_1, ..., u_{s_j} \in U : w = \phi_j(u_1, ..., u_{s_j})\}$$

If the minimum is taken over the empty set, then we set $\gamma(w, U) = \infty$. If $w \in U$, then $\gamma(w, U) = 0$.

Remark 1.3. *(For programmers)* You can always achieve $\gamma(w, U) = 0$ by adding the function $\phi_0 = \text{id}$ to the structure with $\text{¢}(\phi_0) = 0$.

Definition 1.4 (Computation). *1. A sequence $\beta = (w_1, ..., w_m)$ of elements in M is a computation with input $X \subseteq M$ if:*

$$\forall j \leq m: \quad w_j \in X \ \vee \ \gamma(w_j, V_j) < \infty \quad \textit{where } V_j = \{w_1, ..., w_{j-1}\}$$

2. β computes a set $Y \in \mathrm{fin}(M)$ if in addition $Y \subseteq \{w_1, ..., w_m\}$.

3. The costs of β are $\Gamma(\beta, X) \stackrel{Def}{=} \sum_{j=1}^{m} \gamma(w_j, V_j)$.

In a computation, every w_i can be computed from elements previously computed, i.e, elements in V_j or from elements in X ("inputs").

Definition 1.5 (Complexity). *Complexity of Y given X is defined by*

$$C(Y, X) \stackrel{Def}{=} min\{\Gamma(\beta, X) \mid \beta \textit{ computes } Y \textit{ from } X\}.$$

The complexity of a set Y is nothing but the cost of a cheapest computation that computes Y.

Notation. 1. If we compute only one element y, we will write $C(y, X)$ instead of $C(\{y\}, X)$ and so on.

2. If $X = \emptyset$ of X is clear from the context, then we will just write $C(Y)$.

1.3 Examples

The following computation structure will appear quite often in this lecture.

Example 1.6 (Ostrowski measure). Our structure is $M = K(X_1, ..., X_n)$, the field of rational functions in indeterminates $X_1, \ldots, X_n$. We have four (or three) operations of arity 2, namely, multiplication, division, addition, and subtraction. Division is a partial operation which is only defined if the second input is nonzero. If we are only interested in computing polynomials, we might occasionally disallow divisions. For every $\lambda \in K$, there is an operation $\lambda\cdot$ of aritiy 1, the multiplication with the scalar λ. The costs are given by

Operation	Costs
$\cdot$, /	1
+, −	0
$\lambda\cdot$	0

While in nowadays computer chips, multiplication takes about the same number of cycles as addition, Strassen's algorithm and also Karatsuba's algorithm show that this is nevertheless a meaningful way of charging costs.

The complexity induced by the Ostrowski measure will be denoted by $\mathrm{C}^{*/}$, if we allow divisions, or C^*, if we disallow divisions. In particular, Karatsuba's algorithm yields $\mathrm{C}^{*/}(\{c_0, c_1, c_2\}, \{a_0, a_1, b_0, b_1\}) = 3$. (The lower bound follows from the fact, that c_0, c_1, c_2 are linearly independent over K.)

Example 1.7 (Addition chains)**.** Our structure is $M = \mathbb{N}$ with the following operations:

Operation	Arity	Costs
1	0	0
$+$	2	1

$C(n)$ measures how many additions we need to generate n from 1.

Additions chains are motivated by the problem of computing a power X^n from X with as few multiplications as possible. We have $\log n \leq C(n) \leq 2 \log n$. The lower bound follows from the fact that we can at most double the largest number computed so far with one more additon. The upper bound is the well-known "square and multiply" algorithm. This is an old problem from the 1930s, which goes back to Scholz [Sch37] and Brauer [Bra39], but quite some challenging questions still remain open.

Research problem 1.1. Prove the Scholz-Brauer conjecture:

$$C(2^n - 1) \leq n + C(n) - 1 \qquad \text{for all } n \in \mathbb{N}.$$

Research problem 1.2. Prove Stolarsky's conjecture:

$$C(n) \geq \log n + \log(q(n)) \qquad \text{for all } n \in \mathbb{N},$$

where $q(n)$ is the sum of the bits of the binary expansion of n. (Compared to the exercise, there is a "$-O(1)$" missing.)

Chapter 2

Evaluation of polynomials

Let us start with a simple example, the evaluation of univariate polynomials. Our input are the coefficients $a_0, \ldots, a_n$ of the polynomial and the point x at which we want to evaluate the polynomial. We model them as indeterminates, so our set $M = K_0(a_0, \ldots, a_n, x)$. We are interested in determining $C(f, \{a_0, \ldots, a_n, x\})$ where

$$f = a_0 + a_1 x + ... + a_n x^n \in K_0(a_0, ..., a_n, x).$$

A well known algorithm to compute f is the Horner scheme. We write f as

$$f = ((a_n x + a_{n-1})x + a_{n-2})x + ... + a_0.$$

This representation immediately gives a way to compute f with n multiplications and n additions. We will show that this is best possible. Even if we can make as many additions/subtractions as we want, we still need n multiplications/divisions. And even if we are allowed to perform as many multiplications/divisions as we want, n additions/subtractions are required. In the former case, we will use the well-known Ostrowski measure. In the latter case, we will use the so-called additive completity, denoted by C^+, which is "the opposite" of the Ostrowski model. Here multiplications and divisions are for free but additions and subtractions count.

Operation	Costs	
	$C^{*/}$	C^+
$\cdot$, /	1	0
+, −	0	1
$\lambda\cdot$	0	0
$p \in K_0(x)$	0	0

We will even allow that we can get elements from $K := K_0(x)$ for free (operation with arity zero). So we e.g. can compute arbitrary powers of x at no costs. (This is a special feature of this chapter. In general, this is neither the case under the Ostrowski measure nor in the additive measure.)

Theorem 2.1. Let $a_0, ..., a_n, x$ be indeterminates over K_0 and $f = a_0 + a_1 x + ... + a_n x^n$. Then $C^{*/}(f) \geq n$ and $C^+(f) \geq n$. This is even true if all elements from $K_0(x)$ are free of costs.

2.1 Multiplications

The first statement of Theorem 2.1 is implied by the following lower bound.

Theorem 2.2. Let $K_0 \subseteq K$ be fields, $Z = \{z_1, ..., z_n\}$ be indeterminates and $F = \{f_1, ..., f_m\}$ where $f_\mu = \sum_{\nu=1}^{n} p_{\mu,\nu} z_\nu + q_\mu$ with $p_{\mu,\nu}, q_\mu \in K$, $1 \leq \mu \leq m$. Then $\mathrm{C}^{*/}(F, Z) \geq r - m$ where

$$r = \text{col-rk}_{K_0} \begin{pmatrix} p_{11} & \dots & p_{1n} & 1 & \dots & 0 \\ \vdots & & \vdots & \vdots & \ddots & \vdots \\ p_{m1} & \dots & p_{mn} & 0 & \dots & 1 \end{pmatrix}.$$

We get the first part of Theorem 2.1 from Theorem 2.2 as follows: We set

$$\begin{aligned} K &= K_0(x), \\ z_\nu &= a_\nu, \\ m &= 1, \\ f_1 &= f, \\ p_{1,\nu} &= x^\nu, \qquad 1 \leq \nu \leq n, \\ q_1 &= a_0. \end{aligned}$$

Then $P = (x, x^2, ..., x^n, 1)$ and $\text{col-rk}_{K_0} P = n+1$.[1] We get $\mathrm{C}^{*/}(f_1, \{a_0, ..., a_n\}) \geq n+1-1 = n$ by Theorem 2.2.

Proof. (of Theorem 2.2) The proof is by induction in n.
Induction start ($n = 0$): We have

$$P = \begin{pmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{pmatrix}$$

and therefore, $r = m$. Thus $\mathrm{C}^{*/}(F) \geq 0 = r - m$.
Induction step ($n - 1 \to n$): If $r = m$, then there is nothing to show. Thus we can assume that $r > m$. We claim that in this case, $\mathrm{C}^{*/}(F, Z) \geq 1$. This is due to the fact that the set of all rational function that can be computed with costs zero is

$$W_0 = \{w \in K(z_1, ..., z_m) \mid C(w, Z) = 0\} = K + K_0 z_1 + K_0 z_2 + ... + K_0 z_n.$$

(Cleary, every element in W_0 can be computed without any costs. But W_0 is also closed under all operations that are free of costs.) If $r > m$, then there are μ and i such that $p_{\mu,i} \notin K_0$ and therefore $f_\mu \notin W_0$.

W.l.o.g. K_0 is infinite, because if we replace K_0 by $K_0(t)$ for some indeterminate t, the complexity cannot go up, since every computation over K_0 is certainly a computation over $K_0(t)$. W.l.o.g. $f_\mu \neq 0$ for all $1 \leq \mu \leq m$.

[1] Remember that we are talking about the rank over K_0. And over K_0, pairwise distinct powers of x are linearly independent!

Let $\beta = (w_1, ..., w_\ell)$ be an optimal computation for F and let each $w_\lambda = \dfrac{p_\lambda}{q_\lambda}$ with $p_\lambda, q_\lambda \in K_0[z_1, ..., z_n]$. Let j be minimal such that $\gamma(w_j, V_j) = 1$, where $V_j = \{w_1, \ldots, w_{j-1}\}$. Then there are $u, v \in W_0$ such that

$$w_j = \begin{cases} u \cdot v & \text{or} \\ u/v \end{cases}$$

By definition of W_0, there exist $\alpha_1, ..., \alpha_n \in K_0$, $b \in K$ and $\gamma_1, ..., \gamma_n \in K_0$, $d \in K$ such that

$$u = \sum_{\nu=1}^{n} \alpha_\nu z_\nu + b,$$
$$v = \sum_{\nu=1}^{n} \gamma_\nu z_\nu + d.$$

Because $b \cdot d, b/d \notin W_0$, there is a ν_1 such that $\alpha_{\nu_1} \neq 0$ or there is a ν_2 such that $\gamma_{\nu_2} \neq 0$. W.l.o.g. $\nu_1 = n$ or $\nu_2 = n$.

Now the idea is the following. We define a homomorphism $S : M' \to \bar{M}$ where M' is an appropriate subset of M and $\bar{M} = K[z_1, \ldots, z_{n-1}]$ in such a way that

$$C(S(f_1), \ldots, S(f_m)) \leq C(f_1, \ldots, f_m) - 1$$

Such an S is also called a substitution and the proof technique that we are using is called the substitution method. Then we apply the induction hypothesis to $S(f_1), \ldots, S(f_m)$.

Case 1: $w_j = u \cdot v$. We can assume that $\gamma_n \neq 0$. Our substitution S is induced by

$$z_n \to \frac{1}{\gamma_n}(\underbrace{\lambda}_{\in K_0} - \sum_{\nu=1}^{n-1} \gamma_\nu z_\nu - d)$$
$$z_\nu \to z_\nu \qquad \text{for } 1 \leq \nu \leq n-1.$$

The parameter λ will be choosen later. We have $S(z_n) \in W_0$, so there is a computation $(x_1, \ldots, x_t)$ computing z_n at no costs. In the following, for an element $g \in K(z_1, \ldots, z_n)$, we set $\bar{g} := S(g)$. We claim that the sequence

$$\bar{\beta} = (\underbrace{x_1, \ldots, x_t}_{\text{compute } z_n \text{ for free}}, \bar{w_1}, ..., \bar{w_\ell})$$

is a computation for $\bar{f_1}, \ldots, \bar{f}_{m-1}$, since S is a homomorphism. There are two problems that have to be fixed: First z_n (an input) is replaced by something, namely $\bar{z}_n$, that is not an input. But we compute $\bar{z}_n$ in the beginning. Second, the substitution might cause a "division by zero", i.e., there might be an i such that $\bar{q}_i = 0$ and then $\bar{w}_i = \frac{\bar{p}_i}{\bar{q}_i}$ is not defined. But since q_i considered as an element of $K(z_1, \ldots, z_{n-1})[z_n]$ can only have finitely many zeros, we can choose the parameter λ in such a way that none of the $\bar{q}_i$ is zero. (K_0 is infinite!)

By definition of S,

$$\bar{w_j} = \bar{u} \cdot \underbrace{\bar{v}}_{=\lambda},$$

thus

$$\gamma(\bar{w_j}, \bar{V_j}) = 0.$$

This means that

$$\Gamma(\beta, Z) - 1 \geq \bar{\Gamma}(\bar{\beta}, \bar{Z})$$

and

$$\mathrm{C}^{*/}(F, Z) = \Gamma(\beta, Z) \geq \bar{\Gamma}(\bar{\beta}, \bar{Z}) + 1 \underbrace{\geq}_{\text{I.H.}} \text{col-rk}_{K_0} \bar{P} - m + 1.$$

It remains to estimate $\text{col-rk}_{K_0} \bar{P}$. We have

$$\bar{f}_\mu = \sum_{\nu=1}^{n-1} \bar{p}_{\mu,\nu} z_\nu + \bar{q}_\mu$$

$$\bar{p}_{\mu\nu} = p_{\mu\nu} - \frac{\gamma_\nu}{\gamma_n} p_{\mu n}$$

$$\bar{q}_\mu = q_\mu - \frac{p_{\mu n}}{\gamma_n}(\lambda - d)$$

Thus $\bar{P}$ is obtained from P by adding a K_0-multiple of the nth column to the other ones and then deleting the nth column. Therefore, $\text{col-rk}_{K_0} \bar{P} \geq r - 1$ and $\mathrm{C}^{*/}(F, Z) \geq r - m$.

Case 2: $w_j = \dfrac{u}{v}$. If $\gamma_n \neq 0$, then $\bar{v} = \lambda \in K_0$ and the same substitution as in the first case works. If $\gamma_\nu = 0$ for all ν, then $v = d$ and $\alpha_n \neq 0$. Now we substitute

$$z_n \mapsto \frac{1}{\alpha_n}(\lambda d - \sum_{\nu=1}^{n-1} \alpha_\nu z_\nu - b)$$
$$z_\nu \mapsto z_\nu \qquad \text{for } 1 \leq \nu \leq n-1$$

Then $\bar{u} = \lambda d$ and $\bar{w}_j = \dfrac{\bar{u}}{\bar{v}} = \lambda \in K_0$. We can now proceed as in the first case. □

2.1.1 Further Applications

Here are two other applications of Theorem 2.2.

Several polynomials

We can also look at the evaluation of several polynomials at one point x, i.e, at the complexity of

$$f_\mu(x) = \sum_{\nu=0}^{n_\mu} a_{\mu\nu} x^\nu, \qquad 1 \leq \mu \leq m.$$

Here the matrix P looks like

$$P = \left(\begin{array}{cccccccc|ccc} x & x^2 & \dots & x^{n_1} & 0 & \dots & & & 1 & & 0 \\ 0 & & & & x & x^2 & \dots & x^{n_2} & \dots & \ddots & \\ \vdots & & & & & & & & \ddots & 0 & 1 \end{array} \right)$$

and we have $\text{col-rk}_{K_0} P = n_1 + n_2 + \ldots + n_m + m$. Thus

$$\mathrm{C}^{*/}(f_1, \ldots, f_m) \geq n_1 + n_2 + \ldots + n_m,$$

that is, evaluating each polynomial using the Horner scheme is optimal. On the other hand, if we want to evaluate one polynomial at several points, this can be done much faster, see [BCS97].

Matrix vector multiplication

Here, we consider the polynomials $f_1, \ldots, f_m$ given by

$$\begin{pmatrix} a_{11} & \ldots & a_{1k} \\ \vdots & & \vdots \\ a_{m1} & \ldots & a_{mk} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix}$$

The matrix P is given by

$$P = \left(\begin{array}{ccc|ccc} x_1\, x_2\, \ldots\, x_k & & 0 & \ldots & 1 & & 0 \\ & \ddots & & & & \ddots & \\ 0 & & x_1\, x_2\, \ldots\, x_k & & 0 & & 1 \end{array} \right).$$

Thus col-rk$_{K_0}(P) = km + m$ and

$$\mathrm{C}^{*/}(f_1, \ldots, f_m) \geq mk.$$

This means that here—opposed to general matrix multiplication—the trivial algorithm is optimal.

2.2 Additions

The second statement of Theorem 2.1 follows from the following theorem. We need the concept of transcendence degree. If we have two fields $K \subseteq L$, then the transcendence degree of L over K, tr-deg$_K(L)$ is the maximum number t of elements $a_1, \ldots, a_t \in L$ such that $a_1, \ldots, a_t$ do not fulfill any algebraic relation over K, that is, there is no t-variate polynomial p with coefficients from K such that $p(a_1, \ldots, a_t) = 0$.[2]

Theorem 2.3. Let K_0 be a field and $K = K_0(x)$. Let $f = a_0 + \ldots + a_n x^n$. Then $\mathrm{C}^+(f) \geq$ tr-deg$_{K_0}(a_0, a_1, \ldots, a_n) - 1$.

Proof. Let $\beta = (w_1, \ldots, w_\ell)$ be a computation that computes f. W.l.o.g. $w_\lambda \neq 0$ for all $1 \leq \lambda \leq \ell$.

We want to characterize the set W_m of all elements that can be computed with m additions. We claim that there are polynomials $g_i(x, z_1, \ldots, z_i)$ and elements $\zeta_i \in K$, $1 \leq i \leq m$ such that

$$W_0 = \{bx^{t_0} \mid t_0 \in \mathbb{Z}, b \in K\}$$
$$W_m = \{bx^{t_0} f_1(x)^{t_1} \ldots f_m(x)^{t_m} \mid t_i \in \mathbb{Z}, b \in K\}$$

where $f_i(x) = g_i(x, z_1, \ldots, z_i) \mid_{z_1 \to \zeta_1, \ldots, z_i \to \zeta_i}$, $1 \leq i \leq m$. The proof of this claim is by induction in m.

Induction start ($m = 0$): clear by construction

Induction step ($m \to m+1$): Let $w_i = u \pm v$ be the last addition/subtraction in our computation with $m+1$ additions/subtractions. u, v can be computed with m addidition/subtractions, therefore $u, v \in W_m$ by the induction hypothesis. This means that

$$w_i = bx^{t_0} f_1(x)^{t_1} \ldots f_m(x)^{t_m} \pm cx^{s_0} f_1(x)^{s_1} \ldots f_m(x)^{s_m}.$$

[2]Note the similarity to dimension of vector spaces. Here the dimension is the maximum number of elements that do not fulfill any linear relation.

W.l.o.g. $b \neq 0$, othwerwise we would add 0. Therefore,

$$w_i = b(x^{t_0} g_1^{t_1} \dots g_m^{t_m} \pm \frac{c}{b} x^{s_0} g_1^{s_1} \dots g_m^{s_m}) \mid_{z_1 \to \zeta_1, \dots, z_m \to \zeta_m}$$

We set

$$g_{m+1} \stackrel{Def}{=} (x^{t_0} g_1^{t_1} \dots g_m^{t_m} \pm z_{m+1} x^{s_0} g_1^{s_1} \dots g_m^{s_m}).$$

Then

$$w_i = b g_{m+1} \mid_{z_1 \to \zeta_1, \dots, z_{m+1} \to \zeta_{m+1}} \quad \text{with} \quad \zeta_{m+1} = \frac{c}{b}.$$

This shows the claim.

Since w_i was the last addition/substraction in β for every $j > i$, w_j can be computed using only multiplications and is therefore in W_{m+1}. Since the g_i depend on $m+1$ variables $z_1, \dots, z_{m+1}$, the transcendence degree of the coefficients of f is at most $m+1$. □

Exercise 2.1. Show that the additive complexity of matrix vector multiplication is $m(k-1)$. ($m \times k$-matrix with a vector of size k, see the specification in the previous section.)

Chapter 3

Bilinear problems

Let K be a field, we will usually call it the field of scalars, and let $M = K(x_1, ..., x_N)$. We will use the Ostrowski measure in the following. We will ask questions of the form

$$\mathrm{C}^{*/}(F) =?$$

where $F = \{f_1, \ldots, f_k\}$ is a set of quadratic forms,

$$f_\kappa = \sum_{\mu,\nu=1}^{N} t_{\kappa\mu\nu} x_\mu x_\nu.$$

Most of the time, we will consider the special case of bilinear forms, that is, our variable are divided in two disjoint sets and only products of one variable from the first set with one variable of the second set appear in f_κ.

The "three dimensional array" $t := (t_{\kappa\mu\nu})_{\kappa=1,\ldots,k;\mu,\nu=1,\ldots,N} \in K^{k\times N\times N}$ is called the tensor corresponding to F. Since $x_\mu x_\nu = x_\nu x_\mu$, there are several tensors that represent the same set F. A tensor s is symmetrically equivalent to t if

$$s_{\kappa\mu\nu} + s_{\kappa\nu\mu} = t_{\kappa\mu\nu} + t_{\kappa\nu\mu} \qquad \text{for all } \kappa,\ \mu,\ \nu.$$

Two tensors describe the same set of quadratic forms if they are symmetrically equivalent.

The two typical problems that we will deal with in the following are:

Matrix multiplication: We are given two $n \times n$-matrices $x = (x_{i,j})$ and $y = (y_{i,j})$ with indeterminates as entries. The entries of xy are given be the well-known quadratic (in fact bilinear) forms

$$f_{ij} = \sum_{k=1}^{n} x_{ik} y_{kj}, \qquad 1 \le i, j \le n.$$

Polynomial multiplication: Here our input consists of two polynomials $p(z) = \sum_{i=0}^{m} a_i z^i$ and $q(z) = \sum_{j=0}^{n} b_j z^j$. The coefficients are again indeterminates over K. The coefficients c_ℓ, $0 \le \ell \le m + n$ of their product pq are given be the bilinear forms

$$c_\ell = \sum_{i+j=\ell} a_i b_j, \qquad 0 \le \ell \le m + n.$$

Figure 3.1 shows the tensor of multiplication of degree 3 polynomials. It is an element of $K^{4\times4\times7}$. Figure 3.2 shows the tensor of 2×2-matrix multiplication. It lives in $K^{4\times4\times4}$.

	a_0	a_1	a_2	a_3
b_0	1	2	3	4
b_1	2	3	4	5
b_2	3	4	5	6
b_3	4	5	6	7

Figure 3.1: The tensor of the multiplication of multiplication of polynomials of degree three. The rows correspond to the entries of the first polynomial, the colums to the entries of the second. The tensors consist of 7 layers. The entries of the tensor are from $\{0,1\}$. The entry ℓ in position (i,j) means that $t_{i,j,\ell} = 1$, i.e. $a_i \cdot b_j$ occurs in c_ℓ.

	$x_{1,1}$	$x_{1,2}$	$x_{2,1}$	$x_{2,2}$
$y_{1,1}$	$(1,1)$		$(2,1)$	
$y_{2,1}$		$(1,1)$		$(2,1)$
$y_{1,2}$	$(1,2)$		$(2,2)$	
$y_{2,2}$		$(1,2)$		$(2,2)$

Figure 3.2: The tensor of 2×2-matrix multiplication. Again, it is $\{0,1\}$-valued. An entry (κ,ν) in the row (κ,μ) and column (μ,ν) means that $x_{\kappa,\mu}y_{\mu,\nu}$ appears in $f_{\kappa,\nu}$.

3.1 Vermeidung von Divisionen

Strassen [Str73] showed that for computing sets of bilinear forms, divisions do not help (provided that the field of scalars is large enough).

For a polynomial $g \in K[x_1,\ldots,x_N]$, $H_j(g)$ denotes the homogenous part of degree j of g, that is, the sum of all monomials of degree j of g.

Theorem 3.1. Let $F_\kappa = \sum_{\mu,\nu=1}^{N} t_{\kappa\mu\nu}x_\mu x_\nu$, $1 \le \kappa \le k$. If $\#K = \infty$ and $\mathrm{C}^{*/}(F) = \ell$ then there is an optimal computation consisting of products

$$P_\lambda = \big(\sum_{i=1}^{N} u_{\lambda i}x_i\big)\big(\sum_{i=1}^{N} v_{\lambda i}x_i\big), \qquad 1 \le \lambda \le \ell$$

such that $F \subseteq \mathrm{lin}_K\{P_1,\ldots,P_\ell\}$. In particular, $\mathrm{C}^*(F) = \mathrm{C}^{*/}(F)$.

Proof. Let $\beta = (w_1,\ldots,w_L)$ be an optimal computation for F, w.l.o.g $0 \notin F$ and $w_i \neq 0$ for all $1 \le i \le L$. Let $w_i = \dfrac{g_i}{h_i}$ with $g_i, h_i \in K[x_1,\ldots,x_N]$, $h_i, g_i \neq 0$.

As a first step, we want to achieve that

$$H_0(g_i) \neq 0 \neq H_0(h_i), \qquad 1 \le i \le L.$$

We substitute

$$x_i \to \bar{x}_i + \alpha_i, \qquad 1 \le i \le N$$

for some $\alpha_i \in K$. Let the resulting computation be $\bar{\beta} = (\bar{w}_1, \ldots, \bar{w}_l)$ where $\bar{w}_i = \dfrac{\bar{g}_i}{\bar{h}_i}$, $\bar{g}_i(\bar{x}_1, \ldots, \bar{x}_N) = g_i(x_1 + \alpha_1, \ldots, x_N + \alpha_N)$ and $\bar{h}_i(\bar{x}_1, \ldots, \bar{x}_N) = h_i(x_1 + \alpha_1, \ldots, x_N + \alpha_N)$. Since $f_\kappa \in \{w_1, \ldots, w_L\}$,

$$\bar{f}_\kappa(\bar{x}_1, \ldots, \bar{x}_N) = f_\kappa(\bar{x}_1 + \alpha_1, \ldots, \bar{x}_N + \alpha_N) \in \{\bar{w}_1, \ldots, \bar{w}_l\}.$$

Because

$$\bar{f}_\kappa(\bar{x}_1, \ldots, \bar{x}_N) = \sum_{\mu,\nu=1}^{N} t_{\kappa\mu\nu}\bar{x}_\mu\bar{x}_\nu = \sum_{\mu,\nu=1}^{N} t_{\kappa\mu\nu}x_\mu x_\nu + \text{ terms of degree} \leq 1,$$

we can extend the computation $\bar{\beta}$ without increasing the costs such that the new computation computes $f_\kappa(x_1, \ldots, x_N)$, $1 \leq \kappa \leq k$. All we have to do is to compute the terms of degree one, which is free of costs, and subtract them from the $\bar{f}_\kappa(\bar{x}_1, \ldots, \bar{x}_N)$, which is again free of costs. We call the resulting computation again $\bar{\beta}$.

By the following well-known fact, we can choose the α_i in such a way that all $H_0(\bar{g}_i) \neq 0 \neq H_0(\bar{h}_i)$, since $H_0(\bar{g}_i) = g_i(\alpha_1, \ldots, \alpha_N)$ and $H_0(\bar{h}_i) = h_i(\alpha_1, \ldots, \alpha_N)$.

Fact 3.2. *For any finite set of polynomials $\phi_1, \ldots, \phi_n$, $\phi_i \neq 0$ for all i, there are $\alpha_1, \ldots, \alpha_N \in K$ such that $\phi_i(\alpha_1, \ldots, \alpha_N) \neq 0$ for all i provided that $\#K = \infty$.*

Next, we substitute

$$\bar{x}_i \to x_i z, \qquad 1 \leq i \leq N$$

Let $\tilde{\beta} = (\tilde{w}_1, \ldots, \tilde{w}_L)$ be the resulting computation. We view the $\tilde{w}_i$ as elements of $K(x_1, \ldots, x_N)[[z]]$, that is, as formal power series in z with rational functions in $x_1, \ldots, x_N$ as coefficients. This is possible, since every $\bar{w}_i = \dfrac{\bar{g}_i}{\bar{h}_i}$. The substitution above transforms $\bar{g}_i$ and $\bar{h}_i$ into the power series

$$\tilde{g}_i = H_0(\bar{g}_i) + H_1(\bar{g}_i)z + H_2(\bar{g}_i)z^2 + \cdots$$
$$\tilde{h}_i = H_0(\bar{h}_i) + H_1(\bar{h}_i)z + H_2(\bar{h}_i)z^2 + \cdots$$

By the fact below, $\tilde{h}_i$ has in inverse in $K(x_1, \ldots, x_N)[[z]]$ because $H_0(\bar{h}_i) \neq 0$. Thus $\tilde{w}_i = \dfrac{\tilde{g}_i}{\tilde{h}_i}$ is an element of $K(x_1, \ldots, x_N)[[z]]$ and we can write it as

$$\tilde{w}_i = c_i + c_i^{'}z + c_i^{''}z^2 + \cdots$$

Fact 3.3. *A formal power series $\sum_{i=0}^{\infty} a_i z^i \in L[[z]]$ is invertible iff $a_0 \neq 0$. Its inverse is given by $\frac{1}{a_0}(1 + q + q^2 + \cdots)$ where $q = -\sum_{i=1}^{\infty} \frac{a_i}{a_0} z^i$.*

Since in the end, we compute a set of quadratic forms, it is sufficient to compute only $\tilde{w}_i$ up to degree two in z. Because c_i and c_i' can be computed for free in the Ostrowski model, we only need to compute c_i'' in every step.

ith step is a multiplication: We have

$$\tilde{w}_i = \tilde{u} \cdot \tilde{v} = (u + u^{'}z + u^{''}z^2 + \ldots)(v + v^{'}z + v^{''}z^2 + \ldots).$$

We can compute

$$c_i'' = \underbrace{\underbrace{u}_{\in K} v''}_{\text{free of costs}} + u'v' + \underbrace{u'' \underbrace{v}_{\in K}}_{\text{free of costs}}.$$

with one bilinear multiplication.
ith step is a division: Here,

$$\tilde{w}_i = \frac{\tilde{u}}{\tilde{v}} = \frac{u + u'z + u''z + \ldots}{1 + v'z + v''z^2 + \ldots} = (u+u'z+u''z^2)(1-(v'z+v''z^2+\ldots)+(v'z+\ldots)^2-(v'z+\ldots)^3).$$

Thus

$$c_i'' = u'' - u'v' - u(-v'' + (v')^2) = u'' - (u' - \underbrace{uv'}_{\text{free of costs}})v' + \underbrace{uv''}_{\text{free of costs}}$$

can be computed with one costing operation. □

3.2 Rank of bilinear problems

Polynomial multiplication and matrix multiplication are bilinear problems. We can separate the variables into two sets $\{x_1, \ldots, x_M\}$ and $\{y_1, \ldots, y_N\}$ and write the quadratic forms as

$$f_\kappa = \sum_{\mu=1}^{M} \sum_{\nu=1}^{N} t_{\kappa\mu\nu} x_\mu y_\nu, \qquad 1 \le \kappa \le k.$$

The tensor $(t_{\kappa\mu\nu}) \in K^{k\times M\times N}$ is unique and we do not need the notion of symmetric equivalence.

Theorem 3.1 tell us that under the Ostrowski measure, we only have to consider products of linear forms. When computing bilinear forms, it is a natural to restrict ourselves to products of the form linear form in $\{x_1, \ldots, x_M\}$ times a linear form in $\{y_1, \ldots, y_N\}$.

Definition 3.4. *The minimal number of products*

$$P_\lambda = \big(\sum_{\mu=1}^{M} u_{\lambda\mu} x_\mu\big)\big(\sum_{\nu=1}^{N} v_{\lambda\nu} y_\nu\big), \qquad 1 \le \lambda \le \ell$$

such that $F \subseteq \operatorname{lin}\{P_1, \ldots, P_l\}$ is called rank of $F = \{F_1, \ldots, F_k\}$ or bilinear complexity of F. We denote it by $R(F)$.

We can define the rank in terms of tensors, too. Let $t = (t_{\kappa,\mu,\nu})$ be the tensor of F as above. We have

$$\begin{aligned} R(F) \le \ell \Leftrightarrow\ & \text{there are linear forms } u_1, \ldots, u_\ell \text{ in } x_1, \ldots, x_M \\ & \text{and } v_1, \ldots, v_\ell \text{ in } y_1, \ldots, y_N \text{ such that } F \subseteq \operatorname{lin}\{u_1v_1, \ldots, u_\ell v_\ell\} \\ \Leftrightarrow\ & \text{there are } w_{\lambda\kappa} \in K,\ 1 \le \lambda \le \ell,\ 1 \le \kappa \le k \\ & \text{such that } f_\kappa = \sum_{\lambda=1}^{l} w_{\lambda\kappa} u_\lambda v_\lambda = \sum_{\lambda=1}^{\ell} w_{\lambda\kappa} \big(\sum_{\mu=1}^{M} u_{\lambda\mu} x_\mu\big)\big(\sum_{\nu=1}^{N} v_{\lambda\nu} y_\nu\big),\ 1 \le \kappa \le k. \end{aligned}$$

Comparing coefficients, we get

$$t_{\kappa\mu\nu} = \sum_{\lambda=1}^{l} w_{\lambda\kappa} u_{\lambda\mu} v_{\lambda\nu}, \qquad 1 \le \kappa \le k,\ 1 \le \mu \le M,\ 1 \le \nu \le N.$$

Definition 3.5. *Let $w \in K^k$, $u \in K^M$, $v \in K^N$. The tensor $w \otimes u \otimes v \in K^{k\times M\times N}$ with entry $w_\kappa u_\mu v_\nu$ in position (κ, μ, ν) is called a triad.*

From the calculation above, we get

$R(F) \le \ell \Leftrightarrow$ there are $w_1, \dots w_\ell \in K^k$, $u_1 \dots u_\ell \in K^M$, and $v_1 \dots v_\ell \in K^N$ such that

$$t = (t_{\kappa\mu\nu}) = \sum_{\lambda=1}^{\ell} \underbrace{w_\lambda \otimes u_\lambda \otimes v_\lambda}_{\text{triad}}$$

We define the rank $R(t)$ of a tensor t to be the minimal number of triads such that t is the sum of these triads.[1] To every set of bilinear forms F there is a corresponding tensor t and vice versa. As we have seen, their rank is the same.

Example 3.6 (Complex multiplication)**.** Consider the multiplication of complex number viewed as an $\mathbb{R}$-algebra. Its multiplication is described by the two bilinear forms f_0 and f_1 defined by

$$(x_0 + x_1 i)(y_0 + y_1 i) = \underbrace{x_0y_0 - x_1y_1}_{f_0} + \underbrace{(x_0y_1 + x_1y_0)}_{f_1} i$$

It is clear that $R(f_0, f_1) \le 4$. But also $R(f_0, f_1) \le 3$ holds. Let

$$\begin{aligned} P_1 &= x_0y_0, \\ P_2 &= x_1y_1, \\ P_3 &= (x_0 + x_1)(y_0 + y_1). \end{aligned}$$

Then

$$\begin{aligned} f_0 &= P_1 - P_2, \\ f_1 &= P_3 - P_1 - P_2. \end{aligned}$$

Multiplicative complexity and rank are linearly related.

Theorem 3.7. Let $F = \{f_1, \dots, f_k\}$ be a set of bilinear forms in variables $\{x_1, \dots, x_M\}$ and $\{y_1, \dots, y_N\}$. Then

$$\mathrm{C}^{*/}(F) \le R(F) \le 2\,\mathrm{C}^{*/}(F).$$

Proof. The first inequality is clear. For the second, assume that $\mathrm{C}^{*/}(F) = \ell$ and consider an optimal computation. We have

$$\begin{aligned} f_\kappa &= \sum_{\lambda=1}^{\ell} w_{\lambda\kappa} \Big(\sum_{\mu=1}^{M} u_{\lambda\mu} x_\mu + \sum_{\nu=1}^{N} u'_{\lambda\nu} y_\nu\Big)\Big(\sum_{\mu=1}^{M} v'_{\lambda\mu} x_\mu + \sum_{\nu=1}^{N} v_{\lambda\nu} y_\nu\Big) \\ &= \sum_{\lambda=1}^{\ell} w_{\lambda\kappa} \Big(\sum_{\mu=1}^{M} u_{\lambda\mu} x_\mu\Big)\Big(\sum_{\nu=1}^{N} v_{\lambda\nu} y_\nu\Big) + \sum_{\lambda=1}^{\ell} w_{\lambda\kappa} \Big(\sum_{\mu=1}^{M} v'_{\lambda\mu} x_\mu\Big)\Big(\sum_{\nu=1}^{N} u'_{\lambda\nu} y_\nu\Big). \end{aligned}$$

[1] Note the similarity to the definition of rank of a matrix. The rank of a matrix M is the minimum number of rank-1 matrices ("dyads") such such that M is the sum of these rank-1 matrices.

The terms of the form $x_i x_j$ and $y_i y_j$ have to cancel each other, since they do not appear in f_κ. □

Example 3.8 (Winograd's algorithm [Win68])**.** Do products that are not bilinear help in for the computation of bilinear forms? Here is an example. We consider the multiplication of $M \times 2$ matrices with $2 \times N$ matrices. Then entries of the product are given by

$$f_{\mu\nu} = x_{\mu 1} y_{1\nu} + x_{\mu 2} y_{2\nu}.$$

Consider the following MN products

$$(x_{\mu 1} + y_{2\nu})(x_{\mu 2} + y_{1\nu}) \qquad \begin{matrix} 1 \le \mu \le M, \\ 1 \le \nu \le N \end{matrix}$$

We can write

$$f_{\mu\nu} = (x_{\mu 1} + y_{2\nu})(x_{\mu 2} + y_{1\nu}) - x_{\mu 1} x_{\mu 2} - y_{1\nu} y_{2\nu},$$

thus a total of $MN + M + N$ products suffice. Setting $M = 2$, we can multiply 2×2 matrices with $2 \times n$ matrices with $3N + 2$ multiplications. For the rank, the best we know is $\lceil 3\frac{1}{2} N \rceil$ multiplications, which we get by repeatedly applying Strassen's algorithm and possibly one matrix-vector multiplication if N is odd.

Waksman [Wak70] showed that if $\operatorname{char} K \neq 2$, then even $MN + M + N - 1$ products suffice. We get that the multiplicative complexity of 2×2 with 2×3 matrix multiplication is ≤ 10. On the other hand, Alekseyev [Ale85] proved that the rank is 11.

Chapter 4

The exponent of matrix multiplication

In the following $\langle k,m,n\rangle : K^{k\times m}\times K^{m\times n}\to K^{k\times n}$ denotes the the bilinear map that maps a $k\times m$-matrix A and an $m\times n$-matrix B to their product AB. Since there is no danger of confusion, we will also use the same symbol for the corresponding tensor and for the set of bilinear forms $\{\sum_{\mu=1}^{m} X_{\kappa\mu}Y_{\mu\nu} \mid 1\le\kappa\le k,\ 1\le\nu\le n\}$.

Definition 4.1. $\omega = \inf\{\beta \mid \mathrm{R}(\langle n,n,n\rangle) \le O(n^\beta)\}$ *is called the exponent of matrix multiplication.*

In the definition of ω above, we only count bilinear products. For the asymptotic growth, it does not matter whether we count all operations or only bilinear products. Let $\tilde{\omega} = \inf\{\beta \mid C(\langle n,n,n\rangle) \le O(n^\beta)\}$ with $\dot{\varsigma}(\pm) = \dot{\varsigma}(*/) = \dot{\varsigma}(\lambda\cdot) = 1$.

Theorem 4.2. $\omega = \tilde{\omega}$, if K is infinite.

Proof. " $\le$ " : $\surd$

" $\ge$ " : From the definition of ω, it follows that

$$\forall\epsilon>0 : \exists m_0>1 : \forall m\ge m_0 : \mathrm{R}(\langle m,m,m\rangle) \le \underbrace{\alpha}_{\text{w.l.o.g. } =1}\ m^{w+\epsilon}$$

Choose such an m. Let $r = \mathrm{R}(\langle m,m,m\rangle)$. To multiply $m^i\times m^i$-matrices we decompose them into blocks of $m^{i-1}\times m^{i-1}$-matrices and apply recursion. Let $A(i)$ be the number of arithmetic operations for the multiplication of $m^i\times m^i$-matrices with this approach. We obtain

$$A(i) \le rA(i-1) + c\ m^{2(i-1)}$$

where c is the number of additions and scalar multiplications that are performed by the chosen

bilinear algorithm for $\langle m, m, m\rangle$ with r bilinear multiplications. Expanding this, we get

$$\begin{aligned}
A(i) &\le r^i A(0) + cm^{2(i-1)} \left(\sum_{j=0}^{i-2} \frac{r^j}{m^{2j}} \right) \\
&= r^i A(0) + c\ m^{2(i-1)} \frac{\left(\frac{r}{m^2}\right)^{i-1} - 1}{\frac{r}{m^2} - 1} \\
&= r^i A(0) + c\ m^2\ \frac{r^{i-1} - m^{2(i-1)}}{r - m^2} \\
&= \underbrace{\left(A(0) + \frac{c\ m^2}{r(r-m^2)} \right)}_{\text{constant}} r^i - \frac{c}{r-m^2}\, m^2.
\end{aligned}$$

We have $C(\langle n', n', n'\rangle) \le C(\langle n, n, n\rangle)$ if $n' \le n$. (Recall that we can eliminate divisions.) Therefore,

$$\begin{aligned}
C(\langle n, n, n\rangle) &\le C(\left\langle m^{\lceil \log_m n \rceil}, m^{\lceil \log_m n \rceil}, m^{\lceil \log_m n \rceil} \right\rangle) \\
&\le A(\lceil \log_m n \rceil) \\
&= O(r^{\lceil \log_m n \rceil}) \\
&= O(r^{\log_m n}) \\
&= O(n^{\log_m r}).
\end{aligned}$$

Since $r \le m^{\omega+\epsilon}$, we have $\log_m r \le \omega + \epsilon$. Therefore,

$$C(\langle n, n, n\rangle) = O(n^{\log_m r}) = O(n^{\omega+\epsilon})$$

and

$$\tilde{\omega} \le \omega + \epsilon \qquad \text{for all } \epsilon > 0.$$

This means $\tilde{\omega} = \omega$, since $\tilde{\omega}$ is an infimum. □

To prove good upper bounds for ω, we introduce some operation on tensors and analyze the behavior of the rank under these operations.

4.1 Permutations (of tensors)

Let $t \in K^{k \times m \times n}$ and $t = \sum_{j=1}^{r} t_j$ with triads $t_j = a_{j1} \otimes a_{j2} \otimes a_{j3}$, $1 \le j \le r$. Let $\pi \in S_3$. For a triad t_j, let $\pi t_j = a_{j\pi^{-1}(1)} \otimes a_{j\pi^{-1}(2)} \otimes a_{j\pi^{-1}(3)}$ and $\pi t = \sum_{j=1}^{r} \pi t_j$. πt is well-defined. To see this, let $t = \sum_{i=1}^{s} b_{i1} \otimes b_{i2} \otimes b_{i3}$ be a second decomposition of t. We claim that

$$\sum_{j=1}^{r} a_{j\pi^{-1}(1)} \otimes a_{j\pi^{-1}(2)} \otimes a_{j\pi^{-1}(3)} = \sum_{i=1}^{s} b_{i\pi^{-1}(1)} \otimes b_{i\pi^{-1}(2)} \otimes b_{i\pi^{-1}(3)}.$$

Let $a_{j1} = (a_{j11}, \ldots, a_{j1k})$ and $b_{i1} = (b_{i11}, \ldots, b_{i1k})$ and let a_{j2}, a_{j3}, b_{i2}, and b_{i3} be given analogously.

We have

$$t_{e_1e_2e_3} = \sum_{j=1}^{r} a_{j1e_1} \otimes a_{j2e_2} \otimes a_{j3e_3} = \sum_{i=1}^{s} b_{i1e_1} \otimes b_{i2e_2} \otimes b_{i3e_3}.$$

Thus

$$\begin{aligned}\pi t_{e_1e_2e_3} &= \sum_{j=1}^{r} a_{j\pi^{-1}(1)e_{\pi^{-1}(1)}} \otimes a_{j\pi^{-1}(2)e_{\pi^{-1}(2)}} \otimes a_{j\pi^{-1}(3)e_{\pi^{-1}(3)}} \\ &= \sum_{i=1}^{s} b_{i\pi^{-1}(1)e_{\pi^{-1}(1)}} \otimes b_{i\pi^{-1}(2)e_{\pi^{-1}(2)}} \otimes b_{i\pi^{-1}(3)e_{\pi^{-1}(3)}}.\end{aligned}$$

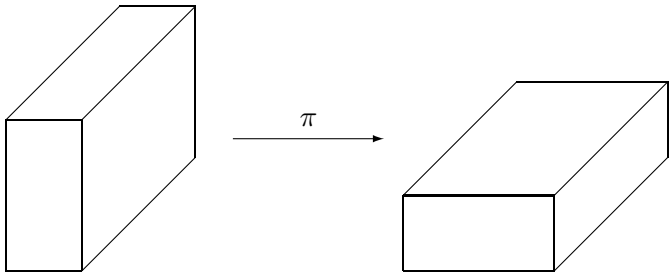

Figure 4.1: Permutation of the dimensions

The proof of the following lemma is obvious.

Lemma 4.3. $\mathrm{R}(t) = \mathrm{R}(\pi t)$.

Instead of permuting the dimensions, we can also permute the slices of a tensor. Let $t = (t_{ijl}) \in K^{k \times m \times n}$ and $\sigma \in S_k$. Then, for $t' = (t_{\sigma(i)jl})$, $\mathrm{R}(t') = \mathrm{R}(t)$.

More general, let $A : K^k \to K^{k'}$, $B : K^m \to K^{m'}$, and $C : K^n \to K^{n'}$ be homomorphisms. Let $t = \sum_{j=1}^{r} t_j$ with triads $t_j = a_{j1} \otimes a_{j2} \otimes a_{j3}$. For a triad t_j, we set

$$(A \otimes B \otimes C)t_j = A(a_{j1}) \otimes B(a_{j2}) \otimes C(a_{j3})$$

and

$$(A \otimes B \otimes C)t = \sum_{j=1}^{r} (A \otimes B \otimes C)t_j.$$

Like above, be looking at a particular entry of t, it is easy to see that this is well-defined.

The proof of the following lemma is again obvious.

Lemma 4.4. $R((A \otimes B \otimes C)t) \leq R(t)$.

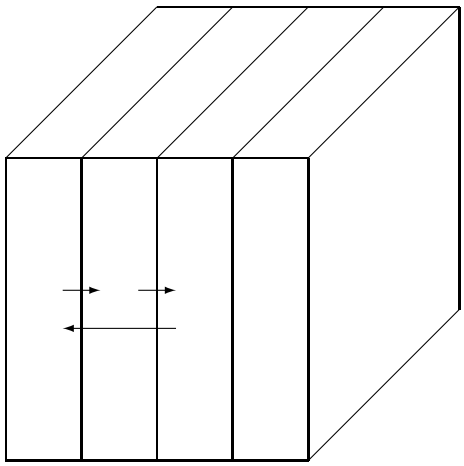

Figure 4.2: Permutation of the slices

Equality holds if A, B, and C are isomorphisms.

How does the tensor of matrix multiplication look like? Recall that the bilinear forms are given by

$$Z_{\kappa\nu} = \sum_{\mu=1}^{m} X_{\kappa\mu} Y_{\mu\nu}, \quad 1 \le \kappa \le k, \;\; 1 \le \nu \le n.$$

The entries of the corresponding tensor

$$(t_{\kappa\bar{\mu},\mu\bar{\nu},\nu\bar{\kappa}}) = t \in K^{(k\times m)\times(m\times n)\times(n\times k)}$$

are given by

$$t_{\kappa\bar{\mu},\mu\bar{\nu},\nu\bar{\kappa}} = \delta_{\bar{\kappa}\kappa}\delta_{\bar{\mu}\mu}\delta_{\bar{\nu}\nu}$$

where δ_{ij} is Kronecker's delta. (Here, each dimension of the tensor is addressed with a two-dimensional index, which reflects the way we number the entries of matrices. If you prefer it, you can label the entries of the tensor with indices from $1, \dots km$, $1, \dots mn$, and $1, \dots, nk$. We also "transposed" the indices in the third slice, to get a symmetric view of the tensor.)

Let $\pi = (123)$. Then for $\pi t =: t' \in K^{(n\times k)\times(k\times m)\times(m\times n)}$, we have

$$\begin{aligned} t'_{\nu\bar{\kappa},\kappa\bar{\mu},\mu\bar{\nu}} &= \delta_{\bar{\nu}\nu}\delta_{\bar{\kappa}\kappa}\delta_{\bar{\mu}\mu} \\ &= \delta_{\bar{\kappa}\kappa}\delta_{\bar{\mu}\mu}\delta_{\bar{\nu}\nu} \\ &= t_{\kappa\bar{\mu},\mu\bar{\nu},\nu\bar{\kappa}} \end{aligned}$$

Therefore,

$$\mathrm{R}(\langle k,m,n\rangle) = \mathrm{R}(\langle n,k,m\rangle) = \mathrm{R}(\langle m,n,k\rangle)$$

Now, let $t'' = (t_{\mu\bar{\kappa},\nu\bar{\mu},\bar{\kappa}\nu})$. We have $R(t) = R(t'')$, since permuting the "inner" indices corresponds to permuting the slices of the tensor.

Next, let $\pi = (12)(3)$. Let $\pi t'' =: t''' \in K^{(n\times m)\times(m\times k)\times(k\times n)}$. We have,

$$\begin{aligned} t'''_{\nu\bar{\mu},\mu\bar{\kappa},\kappa\bar{\nu}} &= \delta_{\mu,\bar{\mu}}\delta_{\kappa,\bar{\kappa}}\delta_{\nu,\bar{\nu}} \\ &= t_{\bar{\kappa}\mu,\bar{\mu}\nu,\bar{\nu}\kappa}. \end{aligned}$$

Therefore,

$$\mathrm{R}(\langle k,m,n\rangle) = \mathrm{R}(\langle n,m,k\rangle).$$

The second transformation corresponds to the well-known fact that $AB = C$ implies $B^T A^T = C^T$.

4.2 Products and sums

Let $t \in K^{k\times m\times n}$ and $t' \in K^{k'\times m'\times n'}$. The direct sum of t and t', $s := t \oplus t' \in K^{(k+k')\times(m+m')\times(n+n')}$, is defined as follows:

$$s_{\kappa\mu\nu} = \begin{cases} t_{\kappa\mu\nu} & \text{if } 1 \le \kappa \le k,\ 1 \le \mu \le m,\ 1 \le \nu \le n \\ t'_{\kappa-k,\mu-m,\nu-n} & \text{if } k+1 \le \kappa \le k+k',\ m+1 \le \mu \le m+m',\ n+1 \le \nu \le n+n' \\ 0 & \text{otherwise} \end{cases}$$

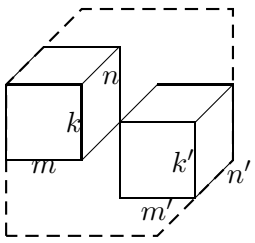

Figure 4.3: Sum of two tensors

Lemma 4.5. $\mathrm{R}(t \oplus t') \le \mathrm{R}(t) + \mathrm{R}(t')$

Proof. Let $t = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i$ and $t' = \sum_{i=1}^{r} u'_i \otimes v'_i \otimes w'_i$. Let

$$\begin{aligned} \hat{u}_i &= (\underbrace{u_{i1},\cdots,u_{i_k}}_{u_i},\underbrace{0,\cdots,0}_{k'}) \quad \text{and} \\ \hat{u}'_i &= (\underbrace{0,\cdots,0}_{k},\underbrace{u'_{i1},\cdots,u'_{ik}}_{u'_i}). \end{aligned}$$

and define $\hat{v}_i$, $\hat{w}_i$ and $\hat{v}'_i$, $\hat{w}'_i$ analogously. And easy calculation shows that

$$t \oplus t' = \sum_{i=1}^{r} \hat{u}_i \otimes \hat{v}_i \otimes \hat{w}_i + \sum_{j=1}^{r'} \hat{u}'_i \otimes \hat{v}'_i \otimes \hat{w}'_i,$$

which proves the lemma. □

Research problem 4.1. (Strassen's additivity conjecture) Show that for all tensors t and t', $R(t \oplus t') = R(t) + R(t')$, that is, equality always holds in the lemma above.

The tensor product $t \otimes t' \in K^{kk' \times mm' \times nn'}$ of two tensors $t \in K^{k \times m \times n}$ and $t' \in K^{k' \times m' \times n'}$ is defined by

$$t \otimes t' = \left(t_{\kappa\mu\nu}\ t'_{\kappa'\mu'\nu'}\right)_{\substack{1 \le \kappa \le k, 1 \le \kappa' \le k' \\ 1 \le \mu \le m, 1 \le \mu' \le m' \\ 1 \le \nu \le n, 1 \le \nu' \le n'}}$$

It is very convenient to use double indices κ, κ' to "address" the slices $1, \ldots, kk'$ of the tensor product. The same is true for the other two dimensions.

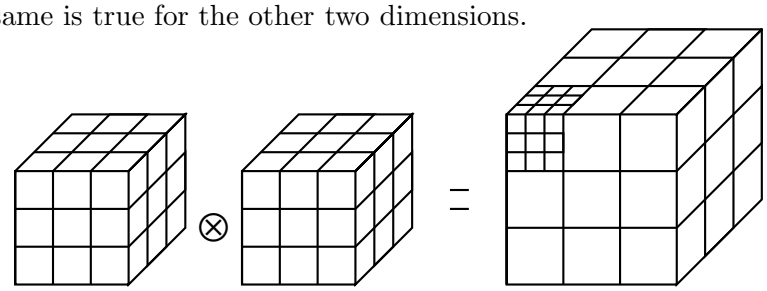

Figure 4.4: Product of two tensors

Lemma 4.6. $\mathrm{R}(t \otimes t') \le \mathrm{R}(t)\,\mathrm{R}(t')$.

Proof. Let $t = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i$ and $t' = \sum_{i=1}^{r'} u'_i \otimes v'_i \otimes w'_i$. Let $u_i \otimes u'_j \overset{Def}{=} (u_{i\kappa} u'_{j\kappa'}) \in K^{kk'}$ In the same way we define $v_i \otimes v'_j$, $w_i \otimes w'_j$. We have

$$(u_i \otimes u'_j) \otimes (v_i \otimes v'_j) \otimes (w_i \otimes w'_j) = (u_{i\kappa} u'_{j\kappa'} \cdot v_{i\mu} v'_{j\mu'} \cdot w_{i\nu} w'_{j\nu'})_{\substack{1 \le \kappa \le k, 1 \le \kappa' \le k' \\ 1 \le \mu \le m, 1 \le \mu' \le m' \\ 1 \le \nu \le n, 1 \le \nu' \le n'}}$$

$$\in K^{kk' \times mm' \times nn'} \cong K^{(k \times k') \times (m \times m') \times (n \times n')}$$

and

$$\sum_{i=1}^{r} \sum_{j=1}^{r'} (u_i \otimes u'_j) \otimes (v_i \otimes v'_j) \otimes (w_i \otimes w'_j) = (\sum_{i=1}^{r} \sum_{j=1}^{r'} u_{i\kappa} u'_{j\kappa'} v_{i\mu} v'_{j\mu'} w_{i\nu} w'_{i\nu'})_{\substack{1 \le \kappa \le k, 1 \le \kappa' \le k' \\ 1 \le \mu \le m, 1 \le \mu' \le m' \\ 1 \le \nu \le n, 1 \le \nu' \le n'}}$$

$$= \Big(\underbrace{(\sum_{i=1}^{r} u_{i\kappa} v_{i\mu} w_{i\nu})}_{t_{\kappa\mu\nu}} \cdot \underbrace{(\sum_{j=1}^{r'} u'_{j\kappa} v'_{j\mu} w'_{j\nu'})}_{t'_{\kappa'\mu'\nu'}} \Big)_{\substack{1 \le \kappa \le k, 1 \le \kappa' \le k' \\ 1 \le \mu \le m, 1 \le \mu' \le m' \\ 1 \le \nu \le n, 1 \le \nu' \le n'}}$$

$$= t \otimes t',$$

which proves the lemma. □

For the tensor product of matrix multiplications, we have

$$\begin{aligned} \langle k, m, n \rangle \otimes \langle k', m', n' \rangle &= (\delta_{\kappa\bar{\kappa}} \delta_{\mu\bar{\mu}} \delta_{\nu\bar{\nu}} \delta_{\kappa'\bar{\kappa}'} \delta_{\mu'\bar{\mu}'} \delta_{\nu'\bar{\nu}'}) \\ &= (\delta_{\kappa\bar{\kappa}} \delta_{\kappa'\bar{\kappa}'} \delta_{\mu\bar{\mu}} \delta_{\mu'\bar{\mu}'} \delta_{\nu\bar{\nu}} \delta_{\nu'\bar{\nu}'}) \\ &= \left(\delta_{(\kappa,\kappa'),(\bar{\kappa},\bar{\kappa}')} \delta_{(\mu,\mu'),(\bar{\mu},\bar{\mu}')} \delta_{(\nu,\nu'),(\bar{\nu},\bar{\nu}')}\right) \\ &= \langle kk', mm', nn' \rangle \end{aligned}$$

Thus, the tensor product of two matrix tensors is a bigger matrix tensor. This corresponds to the well known identity $(A \otimes B)(A' \otimes B') = (AA' \otimes BB')$ for the Kronecker product of matrices. (Note that we use quadruple indices to address the entries of the Kronecker products and also of the slices of of $\langle k, m, n\rangle \otimes \langle k', m', n'\rangle$.)

Using this machinery, we can show that whenever we can multiply matrices of a fixed format efficiently, then we get good bounds for ω.

Theorem 4.7. If $\mathrm{R}(\langle k, m, n\rangle) \leq r$, then $\omega \leq 3 \cdot \log_{kmn} r$.

Proof. If $\mathrm{R}(\langle k, m, n\rangle) \leq r$, then $\mathrm{R}(\langle n, k, m\rangle) \leq r$ and $\mathrm{R}(\langle m, n, k\rangle) \leq r$. Thus

$$\mathrm{R}(\underbrace{\langle k, m, n\rangle \otimes \langle n, k, m\rangle \otimes \langle m, n, k\rangle}_{=\langle kmn, kmn, kmn\rangle}) \leq r^3$$

and, with $N = kmn$,

$$\mathrm{R}(\langle N^i, N^i, N^i\rangle \leq r^{3i} = (N^{3log_N r})^i = (N^i)^{3log_N r}$$

for all $i \geq 1$. Therefore, $\omega \leq 3 \log_N r$. □

Example 4.8 (Matrix tensor of small formats)**.** What do we know about the rank of matrix tensors of small formats?

- $\mathrm{R}(\langle 2, 2, 2\rangle) \leq 7 \implies \omega \leq 3 \cdot \log_{2^3} 7 = \log_2 7 \approx 2.81$
- $\mathrm{R}(\langle 2, 2, 3\rangle) \leq 11$. (This is achieved by doing Strassen once and one trivial matrix-vector product.) This gives a worse bound than 2.81. A lower bound of 11 is shown by [Ale85].
- $14 \leq \mathrm{R}(\langle 2, 3, 3\rangle) \leq 15$, see [BCS97] for corresponding references.
- $19 \leq \mathrm{R}(\langle 3, 3, 3\rangle) \leq 23$. The lower bound is shown in [Blä03], the upper bound is due to Laderman [Lad76]. (We would need ≤ 21 to get an improvement.)
- $\mathrm{R}(\langle 70, 70, 70\rangle) \leq 143.640$ [Pan80]. This gives $\omega \leq 2.80$.

Research problem 4.2. What's the complexity of tensor rank? Hastad [] has shown that this problem is NP-complete over $\mathbb{F}_q$ and NP-hard over $\mathbb{Q}$. What upper bounds can we show over $\mathbb{Q}$? Over $\mathbb{R}$, the problem is decidable, since it reduces to quantifier elemination.

Chapter 5

Border rank

Over $\mathbb{R}, \mathbb{C}$, the rank of matrices is semi-continuous. Let

$$\mathbb{C}^{n\times n} \ni A_j \to A = \lim_{j\to\infty} A_j$$

If for all j, $\mathrm{rk}(A_j) \leq r$, then $\mathrm{rk}(A) \leq r$. $\mathrm{rk}(A_j) \leq r$ means all $(r+1)\times(r+1)$ minors vanish. But since minors are continuous functions, all $(r+1)\times(r+1)$ minor of A vanish, too.

The same is not true for 3-dimensional tensors. Consider multiplication of univariate polynomials of degree one modulo X^2:

$$(a_0 + a_1X)(b_0 + b_1X) = a_0b_0 + (a_1b_0 + a_0b_1)X + a_1b_1X^2$$

The tensor corresponding to the two bilinear forms a_0b_0 and $a_1b_0 + a_0b_1$ has rank 3:

1	0
0	0

0	1
1	0

To show the lower bound, we use the substitution method. We first set $a_0 = 0$, $b_0 = 1$. Then we still compute a_1. Thus there is a product that depends on a_1, say one factor is $\alpha a_0 + \underbrace{\beta a_1}_{\neq 0}$. When we replace a_1 by $-\frac{\alpha}{\beta}a_0$, we kill one product. We still compute a_0b_0 and $-\frac{\alpha}{\beta}a_0b_0 + a_0b_1$. Next, set $a_0 = 1$, $b_0 = 0$. Then we still compute b_1. We can kill another product by substituting b_1 as above. After this, we still compute a_0b_0, which needs one product.

However, we can approximate the tensor above by tensors of rank two. Let

$$t(\epsilon) = (1,\epsilon)\otimes(1,\epsilon)\otimes(0,\frac{1}{\epsilon}) + (1,0)\otimes(1,0)\otimes(1,-\frac{1}{\epsilon})$$

$t(\epsilon)$ obviously has rank two for every $\epsilon > 0$. The slices of $t(\epsilon)$ are

1	0
0	0

0	1
1	ϵ

Thus $t(\epsilon) \to t$ if $\epsilon \to 0$.

Bini, Capovani, Lotti and Romani [BCLR79] used this effect to design better matrix multiplication algorithms. They started with the following partial matrix multiplication:

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \begin{pmatrix} y_{11} & \vert & y_{12} \\ y_{21} & \vert & y_{22} \end{pmatrix} = \begin{pmatrix} z_{11} & \vert & z_{12} \\ z_{21} & \vert & \not{\#} \end{pmatrix}$$

where we only want to compute three entries of the result. We have $\mathrm{R}(\{z_{11}, z_{12}, z_{21}\}) = 6$ but we can approximate $\{z_{11}, z_{12}, z_{21}\}$ with only five products.

That the rank is six can be shown using the substitution method. Consider z_{12}. It clearly depends on y_{12}, so there is a product with one factor being $y_{12} + \ell(y_{11}, y_{21}, y_{22})$ where ℓ is a linear form. Substitute $y_{12} \to -\ell(y_{11}, y_{21}, y_{22})$. This substitution only affects z_{12}. After this substitution we still compute $z_{12}^- = x_{11}(-l(y_{11}, y_{21}, y_{22})) + x_{12}y_{22}$. z_{12}^- still depends on y_{22}. Thus we can substitute again $y_{22} \to -\ell'(y_{11}, y_{21})$. This kills two products and we still compute z_{11}, z_{21}.

Consider the following five products:

$$\begin{aligned} p_1 &= (x_{12} + \epsilon x_{22})y_{21} \\ p_2 &= x_{11}(y_{11} + \epsilon y_{12}) \\ p_3 &= x_{12}(y_{12} + y_{21} + \epsilon y_{22}) \\ p_4 &= (x_{11} + x_{12} + \epsilon x_{21})y_{11} \\ p_5 &= (x_{12} + \epsilon x_{21})(y_{11} + \epsilon y_{22}) \end{aligned}$$

We have

$$\begin{aligned} \epsilon z_{11} &= \epsilon p_1 + \epsilon p_2 + O(\epsilon^2) \\ \epsilon z_{12} &= p_2 - p_4 + p_5 + O(\epsilon^2) \\ \epsilon z_{21} &= p_1 - p_3 + p_5 + O(\epsilon^2) \end{aligned}$$

Now we take a second copy of the partial matrix multiplication above, with new variables. With these two copies, we can multiply 2×2-matrices with 2×3-matrices (by identifying some of the variables in the copy). So we can approximate $\langle 2, 2, 3\rangle$ with 10 multiplications. If approximation would be as good as exact computation, then we would get $\omega \leq 2.79...$ out of this.

We will formalize the concept of approximation. Let K be a field and $K[[\epsilon]] =: \hat{K}$. The role of the small quantity ϵ in the beginning of this chapter is now taken by the indeterminate ϵ.

Definition 5.1. *Let $k \in \mathbb{N}$, $t \in K^{k\times m\times n}$.*

1. $\mathrm{R}_h(t) = \{r \mid \exists u_\rho \in K[\epsilon]^k, v_\rho \in K[\epsilon]^m, w_\rho \in K[\epsilon]^n : \sum_{\rho=1}^{r} u_\rho \otimes v_\rho \otimes w_\rho = \epsilon^h t + O(\epsilon^{h+1})\}$
2. $\underline{\mathrm{R}}(t) = \min_h \mathrm{R}_h(t)$, $\underline{\mathrm{R}}(t)$ *is called the* <u>border rank</u> *of t.*

Remark 5.2. *()*

1. $\mathrm{R}_0(t) = \mathrm{R}(t)$

2. $\mathrm{R}_0(t) \geq \mathrm{R}_1(t) \geq ... = \underline{\mathrm{R}}(t)$

3. For R_t it is sufficient to consider powers up to ϵ^h in u_ρ, v_ρ, w_ρ.

Theorem 5.3. Let $t \in K^{k\times m\times n}$, $t' \in K^{k'\times m'\times n'}$. We have

1) $\forall \pi \in S_3 : \mathrm{R}_h(\pi t) = \mathrm{R}_h(t)$.

2) $\mathrm{R}_{\max\{h,h'\}}(t \oplus t') \leq \mathrm{R}_h(t) + \mathrm{R}_{h'}(t')$.

3) $\mathrm{R}_{h+h'}(t \otimes t') \leq \mathrm{R}_h(t) \cdot \mathrm{R}_{h'}(t')$.

Proof. 1) Clear.

2) W.l.o.g. $h \geq h'$. There are approximate computations such that

$$\sum_{\rho=1}^{r} u_\rho \otimes v_\rho \otimes w_\rho = \epsilon^h t + O(\epsilon^{h+1}) \tag{5.1}$$

$$\sum_{\rho=1}^{r'} \epsilon^{h-h'} u'_\rho \otimes v'_\rho \otimes w'_\rho = \epsilon^{\overset{h}{\#}} t' + O(\epsilon^{\overset{h}{\#}+1}) \tag{5.2}$$

Now we can combine these two computations as we did in the case of rank.

3) Let $t = (t_{ijl})$ and $t' = (t'_{i'j'l'})$. We have $t \otimes t' = (t_{ijl} \cdot t'_{i'j'l'}) \in K^{kk'\times mm'\times nn'}$. Take two approximate computations for t and t' as above. Viewed as exact computations over $K[[\epsilon]]$, their tensor product computes over the following:

$$T = \epsilon^h t + \epsilon^{h+1}\hat{t}, \qquad T' = \epsilon^{h'} t' + \epsilon^{h'+1}\hat{t}'$$

with $\hat{t} \in K[\epsilon]^{k\times m\times n}$ and $\hat{t}' \in K[\epsilon]^{k'\times m'\times n'}$. The tensor product of these two computations computes:

$$\begin{aligned} T \otimes T' &= (\epsilon^h t_{ijl} + \epsilon h + 1\hat{t}_{ijl})(\epsilon^{h'} t'_{i'j'l'} + \epsilon^{h'+1}\hat{t}'_{i'j'l'}) \\ &= (\epsilon^{h+h'} t_{ijl} t'_{i'j'l'} + O(\epsilon^{h+h'+1})) \\ &= \epsilon^{h+h'} t \otimes t' + O(\epsilon^{h+h'+1}) \end{aligned}$$

But this is an approximate computation for $t \otimes t'$.

□

The next lemma shows that we can turn approximate computations into exact ones.

Lemma 5.4. *There is a constant c_h such that for all $t : \mathrm{R}(t) \leq c_h R_h(t)$. c_h depends polynomially on h, in particular $c_h \leq \binom{h+2}{2}$.*

Remark 5.5. *(Better bound for c_h)* Over infinite fields, even $c_h = 1 + 2h$ works.

Proof. Let t be a tensor with border rank r and let

$$\sum_{\rho=1}^{r} \underbrace{\left(\sum_{\alpha=0}^{h} \epsilon^\alpha u_{\rho\alpha}\right)}_{\in K[\epsilon]^k} \otimes \left(\sum_{\beta=0}^{h} \epsilon^\beta v_{\rho\beta}\right) \otimes \left(\sum_{\gamma=0}^{h} \epsilon^\gamma w_{\rho\gamma}\right) = \epsilon^h t + O(\epsilon^{h+1})$$

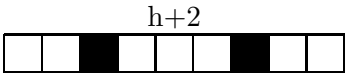

Figure 5.1: By choosing two out of $h+2$ squares, we get a decompsition of h into three pieces. Thus there are $\binom{h+2}{2}$ such decompositions.

The lefthand side of the equation can be rewritten as follows:

$$\sum_{\rho=1}^{r}\sum_{\alpha=0}^{h}\sum_{\beta=0}^{h}\sum_{\gamma=0}^{h}\epsilon^{\alpha+\beta+\gamma}u_{\rho\alpha}\otimes v_{\rho\beta}\otimes w_{\rho\gamma}$$

By comparing the coefficients of ϵ powers, we see that t is the sum of all $u_{\rho\alpha}\otimes v_{\rho\beta}\otimes w_{\rho\gamma}$ with $\alpha+\beta+\gamma=h$. Thus it is only necessary to compute $\binom{h+2}{2}$ products. □

A first attempt to use the results above is to do the following:

$$\begin{array}{rrll} & \mathrm{R}_1(\langle 2,2,3\rangle) & \leq 10 & \\ & \mathrm{R}_1(\langle 3,2,2\rangle) & \leq 10 & \text{(Theorem 5.3.1)} \\ & \mathrm{R}_1(\langle 2,3,2\rangle) & \leq 10 & \\ & \mathrm{R}_3(\langle 12,12,12\rangle) & \leq 1000 & \text{(Theorem 5.3.3)} \\ \Longrightarrow & \mathrm{R}(\langle 12,12,12\rangle) & \leq \binom{3+2}{2}\cdot 1000 = 10\cdot 1000 = 10000 & \end{array}$$

But trivially, $\mathrm{R}(\langle 12,12,12\rangle) \leq 12^3 = 1728$. It turns out that it is better to first "tensor up" and then turn the approximate computation into the exact one.

Theorem 5.6. If $\underline{\mathrm{R}}(\langle k,m,n\rangle) \leq r$ then $\omega \leq 3\log_{kmn} r$.

Proof. Let $N = kmn$ and let $\mathrm{R}_h(\langle k,m,n\rangle) \leq r$.
By Theorem 5.3, we get $\mathrm{R}_{3h}(\langle N,N,N\rangle) \leq r^3$ and $\mathrm{R}_{3hs}(\langle N^s,N^s,N^s\rangle) \leq r^{3s}$ for all s. By Lemma 5.4, this yields $\mathrm{R}(\langle N^s,N^s,N^s\rangle) \leq c_{3hs}r^{3s}$. Therefore,

$$\begin{aligned} \omega &\leq \log_{N^s}(c_{3hs}r^{3s}) \\ &= 3s\log_{N^s}(r) + \log_{N^s}(c_{3hs}) \\ &= 3\log_N(r) + \underbrace{\frac{1}{s}\log_N(\mathrm{poly}(s))}_{\to 0} \end{aligned}$$

Since ω is an infimum, we get $\omega \leq 3\log_N(r)$. □

Corollary 5.7. $\omega \leq 2.79...$

Chapter 6

τ-Theorem

In this chapter, we will consider direct sums of matrix tensors, namely, sums of the form $\mathrm{R}(\langle k,1,n\rangle \oplus \langle 1,m,1\rangle)$. The first summand is the product of vector of length k with a vector of length n, forming a rank-one matrix. The second summand is a scalar product of two vectors of length m.

Remark 6.1. *()*

1. $\mathrm{R}(\langle k,1,n\rangle \oplus \langle 1,m,1\rangle) = k \cdot n + m$
2. $\underline{\mathrm{R}}(\langle k,1,n\rangle) = k \cdot n$ and $\underline{\mathrm{R}}(\langle 1,m,1\rangle) = m$
3. $\underline{\mathrm{R}}(\langle k,1,n\rangle \oplus \langle 1,m,1\rangle) \leq k \cdot n + 1$ with $m = (n-1)(k-1)$.

The first statement is shown by using the substitution method. We first substitute m variables belonging to one vector of $\langle 1,m,1\rangle$. Then we set the variables of the other vector to zero. We still compute $\langle k,1,n\rangle$.

For the second statement, it is sufficient to note that both tensors consist of kn and m linearly independent slices, respectively.

For the third statement, we just prove the case $k = n = 3$. From this, the general construction becomes obvious. So we want to compute $a_i b_j$ for $1 \leq i,j \leq 3$ and $\sum_{\mu=1}^{4} u_\mu v_\mu$. Consider the following products

$$\begin{aligned}
p_1 &= (a_1 + \epsilon u_1)(b_1 + \epsilon v_1)\\
p_2 &= (a_1 + \epsilon u_2)(b_2 + \epsilon v_2)\\
p_3 &= (a_2 + \epsilon u_3)(b_1 + \epsilon v_3)\\
p_4 &= (a_2 + \epsilon u_4)(b_2 + \epsilon v_4)\\
p_5 &= (a_3 - \epsilon u_1 - \epsilon u_3)b_1\\
p_6 &= (a_3 - \epsilon u_2 - \epsilon u_4)b_2\\
p_7 &= a_1(b_3 - \epsilon v_1 - \epsilon v_2)\\
p_8 &= a_2(b_3 - \epsilon v_3 - \epsilon v_4)\\
p_9 &= a_3 b_3
\end{aligned}$$

These nine product obviously compute $a_i b_j$, $1 \leq i,j \leq 3$. Furthermore,

$$\epsilon^2 \sum_{\mu=1}^{4} u_\mu v_\mu = p_1 + \cdots + p_9 - (a_1 + a_2 + a_3)(b_1 + b_2 + b_3).$$

Thus ten products are sufficient to approximate $\langle 3,1,3\rangle \oplus \langle 1,4,1\rangle$.

The second and the third statement together show, that the additivity conjecture is not true for the border rank. We will try to make use of this in the following.

Definition 6.2. *Let $t \in K^{k\times m\times n}$ and $t' \in K^{k'\times m'\times n'}$.*

1. *t is called a restriction of t' if there are homomorphisms $\alpha : K^{k'} \to K^k$, $\beta : K^{m'} \to K^m$, and $\gamma : K^{n'} \to K^n$ such that $t = (\alpha \otimes \beta \otimes \gamma)t'$. We write $t \le t'$.*

2. *t and t' are isomorphic if α, β, γ are isomorphisms ($t \cong t'$).*

In the following, $\langle r\rangle$ denotes the tensor in $K^{r\times r\times r}$ that has a 1 in the positions (ρ,ρ,ρ), $1 \le \rho \le r$, and 0s elsewhere. This tensor corresponds to the r bilinear forms $x_\rho y_\rho$, $1 \le \rho \le r$ (r independent products).

Lemma 6.3. $\mathrm{R}(t) \le r \Leftrightarrow t \le \langle r\rangle$

Proof. "$\Leftarrow$": immediatly from Lemma 4.4.
"$\Rightarrow$": $\langle r\rangle = \sum_{\rho=1}^{r} e_\rho \otimes e_\rho \otimes e_\rho$, where e_ρ is the ρth unit vector. If the rank of t is $\le r$, then we can write t as the sum of r triads,

$$t = \sum_{\rho=1}^{r} u_\rho \otimes v_\rho \otimes w_\rho.$$

We define three homomorphisms:

$$\begin{aligned}
&\alpha \text{ is defined by } e_\rho \mapsto u_\rho;\ \ 1 \le \rho \le r\\
&\beta \text{ is defined by } e_\rho \mapsto v_\rho;\ \ 1 \le \rho \le r\\
&\gamma \text{ is defined by } e_\rho \mapsto w_\rho;\ \ 1 \le \rho \le r
\end{aligned}$$

By construction,

$$(\alpha \otimes \beta \otimes \gamma)\langle r\rangle = \sum_{\rho=1}^{r} \underbrace{\alpha(e_\rho)}_{=u_\rho} \otimes \underbrace{\beta(e_\rho)}_{=v_\rho} \otimes \underbrace{\gamma(e_\rho)}_{=v_\rho} = t$$

which finishs the proof. □

Observation.

1. $t \otimes t' \cong t' \otimes t$
2. $t \otimes (t' \otimes t'') \cong (t \otimes t') \otimes t''$
3. $t \oplus t' \cong t' \oplus t$
4. $t \oplus (t' \oplus t'') \cong (t \oplus t') \oplus t''$
5. $t \otimes \langle 1\rangle \cong t$
6. $t \oplus \langle 0\rangle \cong t$
7. $t \otimes (t' \oplus t'') \cong t \otimes t' \oplus t \otimes t''$.

Above, $\langle 0\rangle$ is the empty tensor in $K^{0\times 0\times 0}$. So the (isomorphism classes of) tensors form a ring. (Remark: If two tensors are isomorphic, then the live in they same space $K^{k\times m\times n}$. If t is any tensor and n is a tensor that is completely filled with zeros, then t is not isomorphic to $t\oplus n$. But from a computational viewpoint, these tensors are the same. So it is also useful to use the wider notion of equivalence. Two tensors t and t' are isomorphic, if there are tensors n and n' completely filled with zeros such that $t\oplus n$ and $t'\oplus n'$ are isomorphic.)

The main result of this chapter is the following theorem due to Schönhage [Sch81]. It is often called τ-theorem in the literature, because the letter τ has a leading role in the original proof. But in our proof, it only has a minor one.

Theorem 6.4. (Schönhage's τ-theorem) If $\underline{\mathrm{R}}(\bigoplus_{i=1}^{p}\langle k_i,m_i,n_i\rangle)\le r$ with $r>p$ then $\omega\le 3\tau$ where τ is defined by

$$\sum_{i=1}^{p}(k_i\cdot m_i\cdot n_i)^{\tau}=r.$$

Notation. Let $f\in\mathbb{N}, t$ be a tensor. $f\odot t \overset{Def}{=} \underbrace{t\oplus\ldots\oplus t}_{f \text{ times}}$.

Lemma 6.5. *If* $\mathrm{R}(f\odot\langle k,m,n\rangle)\le g$, *then* $\omega\le 3\cdot\dfrac{\log\left\lceil\frac{g}{f}\right\rceil}{\log(kmn)}$.

Proof. We first show that for all s, $\mathrm{R}(f\odot\langle k^s,m^s,n^s\rangle)\le\left\lceil\dfrac{g}{f}\right\rceil^s\cdot f$.

The proof is by induction in s: If $s=1$, this is just the assumption of the lemma.
$s\to s+1$: We have

$$\begin{aligned}f\odot\left\langle k^{s+1},m^{s+1},n^{s+1}\right\rangle &= \underbrace{(f\odot\langle k,m,n\rangle)}_{\le\langle g\rangle}\otimes\langle k^s,m^s,n^s\rangle\\ &\le\langle g\rangle\otimes\langle k^s,m^s,n^s\rangle\\ &= g\odot\langle k^s,m^s,n^s\rangle.\end{aligned}$$

Therefore,

$$\begin{aligned}\mathrm{R}(f\odot\left\langle k^{s+1},m^{s+1},n^{s+1}\right\rangle) &\le \mathrm{R}(g\odot\langle k^s,m^s,n^s\rangle)\\ &\le \mathrm{R}(\lceil\tfrac{g}{f}\rceil\cdot f\odot\langle k^s,m^s,n^s\rangle)\\ &= \lceil\tfrac{g}{f}\rceil\cdot\lceil\tfrac{g}{f}\rceil^s\cdot f\\ &= \lceil\tfrac{g}{f}\rceil^{s+1}f\end{aligned}$$

This shows the claim. Now use the claim to proof our lemma: $\mathrm{R}(\langle k^s,m^s,n^s\rangle)\le\lceil\frac{g}{f}\rceil^s\cdot f$ implies

$$\omega\le\frac{3s\log\lceil\frac{g}{f}\rceil+\log(f)\cdot 3}{s\cdot\log(kmn)}=\frac{3\log\lceil\frac{g}{f}\rceil+\overbrace{\log(f)\cdot\frac{3}{s}}^{\to 0 \text{ for } s\to\infty}}{\log(kmn)}.$$

Since ω is an infimum, we get $\omega \leq \dfrac{3\log\lceil \frac{g}{f} \rceil}{\log(kmn)}$. $\square$

Proof of Theorem 6.4. There is an h such that

$$\mathrm{R}_h(\bigoplus_{i=1}^{p} \langle k_i, m_i, n_i \rangle) \leq r.$$

By taking tensor powers and using the fact that the tensor form a ring, we get

$$\mathrm{R}_{hs}\left(\bigoplus_{\sigma_1+\ldots+\sigma_p=s} \frac{s!}{\sigma_1! \cdot \ldots \cdot \sigma_p!} \odot \left\langle \underbrace{\prod_{i=1}^{p} k_i^{\sigma_i}}_{=k'}, \underbrace{\prod_{i=1}^{p} m_i^{\sigma_i}}_{=m'}, \underbrace{\prod_{i=1}^{p} n_i^{\sigma_i}}_{=n'} \right\rangle \right) \leq r^s.$$

k', m', n' depend on $\sigma_1, \ldots, \sigma_p$. Next, we convert the approximate computation into an exact one and get

$$\mathrm{R}\left(\bigoplus_{\sigma_1+\ldots+\sigma_p=s} \frac{s!}{\sigma_1! \cdot \ldots \cdot \sigma_p!} \odot \langle k', m', n' \rangle \right) \leq r^s \cdot \underbrace{c_{hs}}_{\text{polynomial in h and s}}.$$

Define τ by $\sum_{s=\sigma_1+\ldots+\sigma_p} \underbrace{\dfrac{s!}{\sigma_1! \cdot \ldots \cdot \sigma_p!}(k' \cdot m' \cdot n')^\tau}_{=(1)} = r^s$

Fix $\sigma_1, \ldots, \sigma_p$ such that (1) is maximized. Then k', m', and n' are constant. To apply Lemma 6.5, we set

$$\begin{aligned} f &= \frac{s!}{\sigma_1! \cdot \ldots \cdot \sigma_p!} < p^s, \\ g &= r^s \cdot c_{hs}, \\ m &= m, \quad h = h', \quad n = n'. \end{aligned}$$

The number of all $\vec{\sigma}$ with $\sigma_1 + \ldots + \sigma_p = s$ is

$$\binom{s+p-1}{p-1} = \frac{s+p-1}{p-1} \cdot \frac{s+p-2}{p-2} \ldots \leq (s+1)^{p-1}.$$

Thus

$$f \cdot (kmn)^\tau \geq \frac{\sigma^s}{(s+1)^{p-1}}.$$

We get that

$$\left\lceil \frac{g}{f} \right\rceil \leq \frac{r^s \cdot c_{hs}}{f} + 1 \leq (kmn)^\tau \cdot (s+1)^{p-1} \cdot c_{hs}$$

Furthermore,

$$(kmn)^\tau \geq \frac{r^s}{(s+1)^{p-1} f} \geq \frac{r^s}{(s+1)^{p-1} p^s} \tag{6.1}$$

By Lemma 6.5,

$$\begin{aligned}\omega &\leq 3 \cdot \frac{\tau \cdot \log(kmn) + (p-1) \cdot \log(s+1) + \log(c_{hs})}{\log(kmn)} \\ &= 3\tau + \frac{(p-1)\log(s+1) + \log(c_{hs})}{\log(kmn)} \underset{s\to\infty}{\to} 3\tau\end{aligned}$$

because $\log(kmn) \geq s \cdot \underbrace{(\log r - \log p)}_{>0} - O(\log(s))$ by (6.1). □

By using the example at the beginning of this chapter with $k = 4$ and $n = 3$, we get the following bound out of the τ-theorem.

Corollary 6.6. $\omega \leq 2.55$.

Chapter 7

Strassen's Laser Method

Consider the following tensor

$$\mathrm{Str} = \sum_{i=1}^{q} (\underbrace{e_i \otimes e_0 \otimes e_i}_{\langle q,1,1\rangle} + \underbrace{e_0 \otimes e_i \otimes e_i}_{\langle 1,1,q\rangle})$$

This tensor is similar to $\langle 1,2,q\rangle$, only the "directions" of the two scalar products are not the same. But Strassen's tensor can be approximated very efficiently. We have

$$\sum_{i=1}^{q}(e_0+\epsilon e_i)\otimes(e_0+\epsilon e_i)\otimes e_i = \sum_{i=1}^{q} e_0\otimes e_0\otimes e_i + \epsilon\sum_{i=1}^{q}(e_i\otimes e_0\otimes e_i + e_0\otimes e_i\otimes e_i) + O(\epsilon^2)$$

If we subtract the triad $e_0\otimes e_0\otimes\sum_{i=1}^{q} e_i$, we get an approximation of Str. Thus $\underline{\mathrm{R}}(\mathrm{Str}) \leq q+1$.

Definition 7.1. *Let $t \in K^{k\times m\times n}$ be a tensor. Let the sets I_i, J_j, L_ℓ be sucht that:*

$$\begin{array}{ll} 1\leq i\leq p: & I_i\subseteq\{1,\ldots,k\},\\ 1\leq j\leq q: & J_j\subseteq\{1,\ldots,m\},\\ 1\leq l\leq s: & L_\ell\subseteq\{1,\ldots,n\}.\end{array}$$

These sets are called a <u>decomposition D</u> if the following holds:

$$\begin{aligned} I_1\dot\cup I_2\dot\cup\ldots\dot\cup I_p &= \{1,\ldots,k\},\\ J_1\dot\cup J_2\dot\cup\ldots\dot\cup J_q &= \{1,\ldots,m\},\\ L_1\dot\cup L_2\dot\cup\ldots\dot\cup L_s &= \{1,\ldots,n\}.\end{aligned}$$

$t_{I_i,J_j,L_\ell}\in K^{|I_i|\times|J_j|\times|L_\ell|}$ is the tensor that one gets when restricting t to the slices in I_i, J_j, L_ℓ, i.e,

$$(t_{I_i,J_j,L_\ell})_{a,b,c} = t_{\hat a,\hat b,\hat c}$$

Fabian promised to draw a picture.

Figure 7.1: Strassen's tensor

where $\hat{a}$ = the ath largest element in I_i and $\hat{b}$ and $\hat{c}$ are defined analogously. $t_D \in K^{p\times q\times s} = (t_{D,i,j,l})$ is defined by:

$$t_{D,i,j,l} = \begin{cases} 1 & \text{if } t_{I_i,J_j,L_\ell} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Finally, $\operatorname{supp}_D t \overset{Def}{=} \{(i,j,\ell) \mid t_{I_i,J_j,L_\ell} \neq 0\}$.

We can think of giving the tensors an "inner" and an "outer" structure. The t_{I_i,J_j,L_ℓ} are the inner tensor, the tensor t_D is the outer structure. Now we decompose Strassen's tensor and analyse its outer structure: Define D as follows:

$$\begin{cases} \{0\} & \dot\cup & \{1,\dots,q\} & = \{0,\dots,q\} \\ = I_0 & & = I_1 & \\ \{0\} & \dot\cup & \{1,\dots,q\} & = \{0,\dots,q\} \\ = J_0 & & = J_1 & \\ & & \{1,\dots,q\} & = \{1,\dots,q\} \\ & & = L_1 & \end{cases}$$

The outer structure is a matrix tensor,

$$\operatorname{Str}_D = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \langle 1,2,1\rangle.$$

The inner structures are matrix tensors, too,

$$\operatorname{Str}_{I_i,J_j,L_\ell} \in \{\langle q,1,1\rangle, \langle 1,1,q\rangle\}, \qquad \text{for all } (i,j,k) \in \operatorname{supp}_D \operatorname{Str}.$$

Lemma 7.2. *Let $\mathcal{T}, \mathcal{T}'$ be sets of tensors. Let $t \in K^{k\times m\times n}$, $t' \in K^{k'\times m'\times n'}$ with decompositions D, D'. Assume that $t_{I_i,J_j,L_l} \in \mathcal{T}$ for all $(i,j,l) \in supp_D t$ and $t'_{I'_i,J'_j,L'_l} \in \mathcal{T}'$ for all $(i,j,l) \in supp_D t'$. Then*

$$\left.\begin{array}{lll} I_i \times I'_{i'} & ,1 \le i \le p & ,1 \le i' \le p' \\ J_j \times J'_{j'} & ,1 \le j \le q & ,1 \le j' \le q' \\ L_l \times L'_{l'} & ,1 \le l \le s & ,1 \le l' \le s' \end{array}\right\} =: D \otimes D'$$

is a decomposition of $t \otimes t'$ such that

$$(t \otimes t')_{D\otimes D'} \cong t_D \otimes t'_{D'}$$

Furthermore

$$(t \otimes t')_{I_i\times I'_{i'},J_j\times J'_{j'},L_l\times L'_{l'}} \in \mathcal{T} \underbrace{\otimes}_{elementwise} \mathcal{T}'$$

for all $(i,j,l) \in supp_D t$ and $(i',j',l') \in supp_{D'} t'$.

Exercise 7.1. Proof the lemma above.

In the same way, we can prove a similar theorem for sums of tensors and for permutation of tensor.

Now take the permutation $\pi = (1\ 2\ 3)$. We have

$$\pi \operatorname{Str}_{\pi D} = \langle 1,1,2\rangle \quad \text{and} \quad \pi^2 \operatorname{Str}_{\pi^2 D} = \langle 2,1,1\rangle$$

Taking the tensor product of these three tensors and using Lemma 7.2, we get:

$$(\operatorname{Str} \otimes \pi \operatorname{Str} \otimes \pi^2 \operatorname{Str})_{D \otimes \pi D \otimes \pi^2 D} \stackrel{Def}{=} \text{Sym-Str} = \langle 2,2,2\rangle$$

with every nonzeor inner tensor in begin an element of $\{\langle k,m,n\rangle \mid k \cdot m \cdot n = q^3\}$

If a tensor t is a restriction of a tensor t', then $\mathrm{R}(t) \leq \mathrm{R}(t')$. It is easy to check that also $\underline{\mathrm{R}}(t) \leq \underline{\mathrm{R}}(t')$. We can generalize restrictions further such that they are still compliant with border rank (but not with rank). Let $A(\epsilon) \in K[\epsilon]^{k \times k'}$, $B(\epsilon) \in K[\epsilon]^{m \times m'}$, $C(\epsilon) \in K[\epsilon]^{n \times n'}$ be polynomial matrices, i.e., matrices whose entries are polynomials in ϵ. For a tensor $t' \in K^{k' \times m' \times n'}$ with a decomposition $t' = \sum_{\rho=1}^{r} u_\rho \otimes v_\rho \otimes w_\rho$, we set

$$(A(\epsilon) \otimes B(\epsilon) \otimes C(\epsilon))t' \stackrel{Def}{=} \sum_{\rho=1}^{r} A(\epsilon)u_\rho \otimes B(\epsilon)v_\rho \otimes C(\epsilon)w_\rho.$$

As before, it is easy to check that this definition is independent of the decomposition and therefore well-defined. is well-defined.

Definition 7.3. *Let $t \in K^{k \times m \times n}$, $t' \in K^{k' \times m' \times n'}$. t is a <u>degeneration</u> of t' if there are $A(\epsilon) \in K[\epsilon]^{k \times k'}$, $B(\epsilon) \in K[\epsilon]^{m \times m'}$, $C(\epsilon) \in K[\epsilon]^{n \times n'}$ and $q \in \mathbb{N}$ such that*

$$\epsilon^q t = (A(\epsilon) \otimes B(\epsilon) \otimes C(\epsilon))t' + O(\epsilon^{q+1}).$$

We will write $t \trianglelefteq_q t'$ or $t \trianglelefteq t'$.

Remark 7.4. *()*$\underline{\mathrm{R}}(t) \leq r \Leftrightarrow t \trianglelefteq \langle r\rangle$

Lemma 7.5. *For all odd n,*

$$\left\langle \left\lceil \frac{3}{4}n^2 \right\rceil \right\rangle \trianglelefteq \langle n,n,n\rangle$$

Furthermore, this degeneration is achieved by a monomial mapping, that is, the matrices $A(\epsilon)$, $B(\epsilon)$, and $C(\epsilon)$ are diagonal matrices with ϵ-powers on the diagonal.

Before we proof this lemma, let us understand what is means. $\underline{\mathrm{R}}(\langle n,n,n\rangle) \leq r$ or equivalently, $\langle n,n,n\rangle \trianglelefteq \langle r\rangle$, means that with r bilinear multiplication, we can "buy" the tensor $\langle n,n,n\rangle$. $\langle \ell\rangle \trianglelefteq \langle n,n,n\rangle$ means, that if we "resell" the tensor $\langle n,n,n\rangle$, then we get ℓ bilinear multiplications back.

Proof. Let $n = 2\nu + 1$. We label rows and colums of the matrices from $-\nu, \ldots, \nu$. We define the matrices A, B, and C by specifying they values on the standard basis of $k^{n \times n}$:

$$\begin{aligned} A: \quad & e_{ij} \quad \to e_{ij} \cdot \epsilon^{i^2+2ij} \\ B: \quad & e_{jk} \quad \to e_{jk} \cdot \epsilon^{j^2+2jk} \\ C: \quad & e_{ki} \quad \to e_{ki} \cdot \epsilon^{k^2+2ki} \end{aligned}$$

so each matrix is a diagonal matrix with ϵ powers on the diagonal.

We have

$$\langle n,n,n\rangle = \sum_{i,j,k=1}^{n} e_{ij}\otimes e_{jk}\otimes e_{ki},$$

thus

$$(A\otimes B\otimes C)\langle n,n,n\rangle = \sum_{i,j,k=1}^{n} \underbrace{\epsilon^{i^2+2ij+j^2+2jk+k^2+2ki}}_{=\epsilon^{(i+j+k)^2}} e_{ij}\otimes e_{jk}\otimes e_{ki}.$$

If $i+j+k=0$ then $\left\{\begin{array}{c} i,k \\ i,j \\ j,k \end{array}\right\}$ determine $\left\{\begin{array}{c} j \\ k \\ i \end{array}\right\}$. So all terms with exponent 0 form a set of independent products. It is easy to see that there are $\geq \frac{3}{4}n^2$ triples (i,j,k) with $i+j+k=0$. □

Now we start with the tensor Sym-Str and the corresponding decomposition Sym-D. Then we take sth tensor power. The outer structure $\text{Sym-Str}^{\otimes s}_{\text{Sym-}D^{\otimes s}}$ is isomorphic to $\langle 2^s,2^s,2^s\rangle$. The nonzero tensors of the inner structure are all of the form $\langle k,m,n\rangle$ with $kmn=q^{3s}$.

We have

$$\left\langle \frac{3}{4}2^{2s}\right\rangle \underbrace{\trianglelefteq}_{\text{Lemma 7.5}} (\text{Sym-Str})^{\otimes s}_{\text{Sym-}D^{\otimes s}}$$

Since the degeneration above is a monomial degeneration, we get, by extending the degeneration to the whole tensor, that a direct sum of $\frac{3}{4}2^s$ matrix tensors $\langle k_i,m_i,n_i\rangle$, $1\leq i\leq \frac{3}{4}2^s$ with $k_im_in_i=q^{3s}$. To this sum, we can apply the τ-theorem and get

$$(q^{3s})^{\tau}\frac{3}{4}2^{2s}\leq (q+1)^{3s}$$
$$q^{3\tau}\underbrace{\sqrt[s]{\frac{3}{4}}}_{\to 1}2^2\leq (q+1)^3.$$

Therefore, $\omega\leq \log_q \frac{(q+1)^3}{4}$. This is minimal for $q=5$ and gives us the result $\omega\leq 2.48$.

Theorem 7.6. (Strassen [Str87]) $\omega\leq 2.48$

Research problem 7.1. What is $\underline{\text{R}}$(Sym-Str)? Is it strictly smaller than $(q+1)^3$.

Bibliography

[Ale85] Valery B. Alekseyev. On the complexity of some algorithms of matrix multiplication. J. Algorithms, 6(1):71–85, 1985.

[BCLR79] Dario Bini, Milvio Capovani, Grazia Lotti, and Francesco Romani. $O(n^{2.7799})$ complexity for matrix multiplication. Inform. Proc. Letters, 8:234–235, 1979.

[BCS97] Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. Algebraic Complexity Theory. Springer, 1997.

[Blä03] Markus Bläser. On the complexity of the multiplication of matrices of small formats. J. Complexity, 19:43–60, 2003.

[Bra39] A. T. Brauer. On addition chains. Bulletin of the American Mathematical Society, 45:736–739, 1939.

[Bsh95] Nader H. Bshouty. On the additive complexity of 2×2-matrix multiplication. Inform. Proc. Letters, 56(6):329–336, 1995.

[Lad76] J. Laderman. A noncommutative algorithm for multiplying 3×3–matrices using 23 multiplications. Bull. Amer. Math. Soc., 82:180–182, 1976.

[Pan80] Victor Ya. Pan. New fast algorithms for matrix multiplication. SIAM J. Comput, 9:321–342, 1980.

[Sch37] Arnold Scholz. Aufgabe 253. Jahresberichte der deutschen Mathematiker-Vereinigung, 47:41–42, 1937.

[Sch81] Arnold Schönhage. Partial and total matrix multiplication. SIAM J. Comput, 10:434–455, 1981.

[Str69] Volker Strassen. Gaussian elimination is not optimal. Numer. Math., 13:354–356, 1969.

[Str73] Volker Strassen. Vermeidung von Divisionen. J. Reine Angew. Math., 264:184–202, 1973.

[Str87] Volker Strassen. Relative bilinear complexity and matrix multiplication. J. Reine Angew. Math., 375/376:406–443, 1987.

[Wak70] A. Waksman. On Winograd's algorithm for inner products. IEEE Trans. Comput., C–19:360–361, 1970.

[Win68] Shmuel Winograd. A new algorithm for inner products. IEEE Trans. Comput., C–17:693–694, 1968.

[Win71] Shmuel Winograd. On multiplication of 2×2–matrices. Lin. Alg. Appl., 4:381–388, 1971.