Московский государственный университет
имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики
Кафедра математической кибернетики

Курсовая работа

«Поиск оптимальных алгоритмов умножения матриц»

студента группы 318

Баканова Артёма Михайловича

Научный руководитель:

д.ф.-м.н., проф.

Алексеев В. Б.

Москва, 2023

Содержание

1 Введение

Умножение матриц является одной из фундаментальных операций в линейной алгебре и широко используется в различных приложениях, таких как машинное обучение, компьютерное зрение и обработка сигналов. Однако, найти оптимальный алгоритм умножения матриц является сложной задачей.

Традиционно сложность алгоритмов умножения матриц оценивается в количестве операций умножения чисел. Это связано с тем, что последняя является наиболее ресурсозатратной. Таким образом, оптимальный алгоритм должен выигрывать у остальных по числу умножений. Так, стандартный алгоритм умножения матрицы $m \times n$ на матрицу $n \times p$

$$z_{ij} = \sum_{k=1}^{n} x_{ik} y_{kj}, \quad i = \overline{1, m},\ j = \overline{1, p} \tag{1}$$

требует n^3 операций умножения, то есть имеет асимптотическую сложность $\mathcal{O}(n^3)$.

Крайне важным шагом в поиске оптимальных алгоритмов стала работа Штрассена [1]. В ней он приводит алгоритм умножения двух квадратных матриц порядка n с асимптотической сложностью $\mathcal{O}(n^{\log_2 7})$. Алгоритм Штрассена основан на том, что для умножения квадратных матриц порядка 2 требуется лишь 7 операций умножения.

2 Постановка задачи

Была выдвинута гипотеза, что существует алгоритм умножения двух квадратных матриц порядка 3, использующий 19 умножений. Необходимо было описать данную задачу в терминах системы уравнений, а затем, используя ряд предположений, перейти к новой системе. Далее требовалось составить программу, вычисляющую последнюю систему в символьном виде.

3 Основная часть

3.1 Переход к системе уравнений

Прежде всего заметим, что стандартный алгоритм умножения матриц (1) представляет собой задачу о вычислении семейства билинейных форм. Запишем её как

$$z_{sr} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{p} a_{(i,j),(k,l),(r,s)} x_{ij} y_{kl}, \quad s = \overline{1,m},\ r = \overline{1,p}, \tag{2}$$

где

$$a_{(i,j),(k,l),(r,s)} = \begin{cases} 1, \text{ если } j = k, l = r, s = i; \\ 0, \text{ иначе.} \end{cases} \tag{3}$$

Пусть для данного семейства существует билинейный алгоритм с мультипликативной сложностью q вида

$$D_1(x,y) = L'_1(x,y) \cdot L''_1(x,y), \quad \ldots \quad, D_q(x,y) = L'_q(x,y) \cdot L''_q(x,y),$$
$$z_{11} = \sum_{d=1}^{q} \gamma_{11}^{d} D_d(x,y), \quad \ldots \quad, z_{mp} = \sum_{d=1}^{q} \gamma_{pm}^{d} D_d(x,y),$$

где

$$L'_1(x,y) = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^{1} x_{ij}, \quad L''_1(x,y) = \sum_{k=1}^{n}\sum_{l=1}^{p} \beta_{kl}^{1} y_{kl};$$
$$\ldots \quad ,$$
$$L'_q(x,y) = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^{q} x_{ij}, \quad L''_q(x,y) = \sum_{k=1}^{n}\sum_{l=1}^{p} \beta_{kl}^{q} y_{kl}.$$

Подставив формулы для $D_d(x,y), L'_d(x,y), L''_d(x,y)$ в выражение для z_{sr}, получим:

$$z_{sr} = \sum_{d=1}^{q} \gamma_{rs}^{d} \left(\sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij}^{d} x_{ij}\right)\left(\sum_{k=1}^{n}\sum_{l=1}^{p} \beta_{kl}^{d} y_{kl}\right). \tag{4}$$

Раскроем скобки в (4) и приравняем к (2). Получим:

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{p} a_{(i,j),(k,l),(r,s)} x_{ij} y_{kl} = \sum_{d=1}^{q}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{p} \alpha_{ij}^{d} \beta_{kl}^{d} \gamma_{rs}^{d} x_{ij} y_{kl}. \tag{5}$$

Учитывая, что матрицы $\|x_{ij}\|$ и $\|y_{kl}\|$ произвольны, и используя (2), окончательно получим:

$$\sum_{d=1}^{q} \alpha_{ij}\beta_{kl}\gamma_{rs} = \begin{cases} 1, \text{ если } j = k, l = r, s = i; \\ 0, \text{ иначе.} \end{cases} \tag{6}$$

Таким образом, алгоритм умножения матрицы $m \times n$ на матрицу $n \times p$, использующий q операций умножения чисел, существует тогда и только тогда, когда разрешима система (6).

3.2 Переход к новой системе

В нашей задаче $m = n = p = 3$, $q = 19$. Будем считать, что i, j, k, l, r, s принимают 3 значения: $0, 1, 2$, а d принимает 19 значений: $0, 1, 2, \ldots, 18$. Тогда система (6) примет вид:

$$\sum_{d=0}^{18} \alpha_{ij}\beta_{kl}\gamma_{rs} = \begin{cases} 1, \text{ если } j = k, l = r, s = i; \\ 0, \text{ иначе.} \end{cases} \tag{7}$$

Она состоит из 729 уравнений и имеет 513 переменных.

Введём квадратные матрицы порядка 3 $A_d = \|\alpha_{ij}^d\|$, $B_d = \|\beta_{kl}^d\|$, $C_d = \|\gamma_{rs}^d\|$, $d = \overline{0, 18}$. Если записать алгоритм Штрассена в терминах матриц A_d, B_d, C_d, получим, что $A_0 = B_0 = C_0 = I$. Потребуем того же и в нашем случае:

$$\alpha_{ij}^0\beta_{kl}^0\gamma_{rs}^0 = \begin{cases} 1, \text{ если } i = j, k = l, r = s; \\ 0, \text{ иначе.} \end{cases}$$

Теперь систему (7) можно записать, как

$$\sum_{d=1}^{18} \alpha_{ij}^d\beta_{kl}^d\gamma_{rs}^d = \begin{cases} -1, \text{ если } i = j, k = l, r = s, \text{ но не выполняется } i = j = k = l = r = s; \\ 1, \text{ если } j = k, l = r, s = i, \text{ но не выполняется } i = j = k = l = r = s; \\ 0, \text{ иначе.} \end{cases} \tag{8}$$

Для удобства все α_{ij}^d домножим на -1 и новые переменные снова обозначим α_{ij}^d. Система примет вид:

$$\sum_{d=1}^{18} \alpha_{ij}^d\beta_{kl}^d\gamma_{rs}^d = \begin{cases} +1, \text{ если } i = j, k = l, r = s, \text{ но не выполняется } i = j = k = l = r = s; \\ -1, \text{ если } j = k, l = r, s = i, \text{ но не выполняется } i = j = k = l = r = s; \\ 0, \text{ иначе.} \end{cases} \tag{9}$$

Неизвестные переменные в системе (9) задаются 18 тройками матриц A_d, B_d, C_d. Наложим дополнительное условие: пусть первые 6 троек произвольны, а остальные 12 получаются из этих шести циклическим сдвигом. То есть из каждой тройки матриц A_d, B_d, C_d, $d = \overline{1,6}$ получаются еще 2 тройки: B_d, C_d, A_d и C_d, A_d, B_d. Тогда система (9) преобразуется к виду:

$$\sum_{d=1}^{6}(\alpha_{ij}^d\beta_{kl}^d\gamma_{rs}^d + \alpha_{kl}^d\beta_{rs}^d\gamma_{ij}^d + \alpha_{rs}^d\beta_{ij}^d\gamma_{kl}^d) =$$

$$\begin{cases} +1, \text{ если } i=j, k=l, r=s, \text{ но не выполняется } i=j=k=l=r=s; \\ -1, \text{ если } j=k, l=r, s=i \text{ но не выполняется } i=j=k=l=r=s; \\ 0, \text{ иначе.} \end{cases} \qquad (10)$$

Уравнения для шестёрок (ij, kl, rs), (kl, rs, ij), (rs, ij, kl) оказываются идентичными. Таким образом, исходные 729 уравнений, кроме 9, для которых шестёрки имеют вид (ij, ij, ij), разбиваются на группы по три. Выбрав по одному уравнению из каждой группы, получим $\frac{729-9}{3} + 9 = 249$ уравнений, содержащих $\frac{513-27}{3} = 162$ переменные.

Продолжим накладывать условия на тройки матриц. На множестве $\{0, 1, 2\}$ рассмотрим всевозможные перестановки:

$$\varphi_1 : \begin{pmatrix} 0 & 1 & 2 \\ 0 & 1 & 2 \end{pmatrix}, \ \varphi_2 : \begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \end{pmatrix}, \ \varphi_3 : \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix},$$
$$\varphi_4 : \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 2 \end{pmatrix}, \ \varphi_5 : \begin{pmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix}, \ \varphi_6 : \begin{pmatrix} 0 & 1 & 2 \\ 2 & 0 & 1 \end{pmatrix}.$$

Пару $\varphi_m(i)\varphi_m(j)$ обозначим, как $\varphi_m(ij)$, $m = \overline{1,6}$. Введем новые переменные в виде 6 квадратных матриц порядка 3:

$$X = \|x_{ij}\|, Y = \|y_{kl}\|, Z = \|z_{rs}\|, A = \|a_{ij}\|, B = \|b_{kl}\|, C = \|c_{rs}\|$$

и положим

$$\alpha_{ij}^1 = x_{\varphi_1(ij)} = x_{ij}, \alpha_{ij}^2 = x_{\varphi_2(ij)},$$
$$\alpha_{ij}^3 = a_{\varphi_3(ij)}, \alpha_{ij}^4 = a_{\varphi_4(ij)}, \alpha_{ij}^5 = a_{\varphi_5(ij)}, \alpha_{ij}^6 = a_{\varphi_6(ij)},$$
$$\beta_{kl}^1 = y_{\varphi_1(kl)} = y_{kl}, \beta_{kl}^2 = y_{\varphi_2(kl)},$$
$$\beta_{kl}^3 = b_{\varphi_3(kl)}, \beta_{kl}^4 = b_{\varphi_4(kl)}, \beta_{kl}^5 = b_{\varphi_5(kl)}, \beta_{kl}^6 = b_{\varphi_6(kl)},$$
$$\gamma_{rs}^1 = z_{\varphi_1(rs)} = z_{rs}, \gamma_{rs}^2 = z_{\varphi_2(rs)},$$
$$\gamma_{rs}^3 = c_{\varphi_3(rs)}, \gamma_{rs}^4 = c_{\varphi_4(rs)}, \gamma_{rs}^5 = c_{\varphi_5(rs)}, \gamma_{rs}^6 = c_{\varphi_6(rs)}.$$

Тогда система (10) примет вид:

$$x_{ij}y_{kl}z_{rs} + x_{kl}y_{rs}z_{ij} + x_{rs}y_{ij}z_{kl}+$$
$$+x_{\varphi_2(ij)}y_{\varphi_2(kl)}z_{\varphi_2(rs)} + x_{\varphi_2(kl)}y_{\varphi_2(rs)}z_{\varphi_2(ij)} + x_{\varphi_2(rs)}y_{\varphi_2(ij)}z_{\varphi_2(kl)}+$$
$$+\sum_{d=3}^{6}(a_{\varphi_d(ij)}b_{\varphi_d(kl)}c_{\varphi_d(rs)} + a_{\varphi_d(kl)}b_{\varphi_d(rs)}c_{\varphi_d(ij)} + a_{\varphi_d(rs)}b_{\varphi_d(ij)}c_{\varphi_d(kl)}) =$$

$$\begin{cases} +1, \text{ если } i = j, k = l, r = s, \text{ но не выполняется } i = j = k = l = r = s; \\ -1, \text{ если } j = k, l = r, s = i \text{ но не выполняется } i = j = k = l = r = s; \\ 0, \text{ иначе.} \end{cases} \tag{11}$$

Важно отметить, что уравнения для шестерок (ij, kl, rs) и $(\varphi_2(ij), \varphi_2(kl), \varphi_2(rs))$ оказываются идентичными. Таким образом, все 249 уравнений, кроме одного, шестёрка которого имеет вид $(11, 11, 11)$, разбиваются на пары. Выбрав по представителю для каждой пары, получим $\frac{249-1}{2}+1 = 125$ уравнений, которые содержат 54 новых переменных. Из неких соображений уменьшим все неизвестные в матрице X в 2 раза и для новых переменных сохраним те же обозначения. Тогда чтобы система (11) оставалась правильной, необходимо все слагаемые в первой и во второй строках домножить на 2. Получаем:

$$2\cdot(x_{ij}y_{kl}z_{rs}+x_{kl}y_{rs}z_{ij}+x_{rs}y_{ij}z_{kl}+$$
$$+x_{\varphi_2(ij)}y_{\varphi_2(kl)}z_{\varphi_2(rs)}+x_{\varphi_2(kl)}y_{\varphi_2(rs)}z_{\varphi_2(ij)}+x_{\varphi_2(rs)}y_{\varphi_2(ij)}z_{\varphi_2(kl)})+$$
$$+\sum_{d=3}^{6}(a_{\varphi_d(ij)}b_{\varphi_d(kl)}c_{\varphi_d(rs)}+a_{\varphi_d(kl)}b_{\varphi_d(rs)}c_{\varphi_d(ij)}+a_{\varphi_d(rs)}b_{\varphi_d(ij)}c_{\varphi_d(kl)})=$$

$$\begin{cases}+1, \text{ если } i=j, k=l, r=s, \text{ но не выполняется } i=j=k=l=r=s;\\ -1, \text{ если } j=k, l=r, s=i \text{ но не выполняется } i=j=k=l=r=s;\\ 0, \text{ иначе.}\end{cases} \tag{12}$$

Обозначим $\varepsilon=-\frac{1}{2}+\frac{\sqrt{3}}{2}i$ – комплексный корень третьей степени из 1. Введём условные записи:

$$(1)=11+00+22,$$
$$(0)=11+\varepsilon\cdot 00+\varepsilon^2\cdot 22,$$
$$(2)=11+\varepsilon^2\cdot 00+\varepsilon\cdot 22,$$
$$(3)=(02+21+10)+(20+01+12),$$
$$(4)=[(02+21+10)-(20+01+12)]\cdot\frac{i}{\sqrt{3}},$$
$$(5)=(02+\varepsilon^2\cdot 21+\varepsilon\cdot 10)+(20+\varepsilon\cdot 01+\varepsilon^2\cdot 12),$$
$$(6)=(02+\varepsilon\cdot 21+\varepsilon^2\cdot 10)+(20+\varepsilon^2\cdot 01+\varepsilon\cdot 12),$$
$$(7)=[(02+\varepsilon\cdot 21+\varepsilon^2\cdot 10)-(20+\varepsilon^2\cdot 01+\varepsilon\cdot 12)]\cdot\frac{i}{\sqrt{3}},$$
$$(8)=[(02+\varepsilon^2\cdot 21+\varepsilon\cdot 10)-(20+\varepsilon\cdot 01+\varepsilon^2\cdot 12)]\cdot\frac{i}{\sqrt{3}}.$$

В заключительный раз введём новые переменные:

$$x_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}x_{ij}, m=\overline{0,8}, y_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}y_{ij}, m=\overline{0,8}, z_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}z_{ij}, m=\overline{0,8},$$
$$a_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}a_{ij}, m=\overline{0,8}, b_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}b_{ij}, m=\overline{0,8}, c_m=\sum_{i=0}^{2}\sum_{j=0}^{2}t_{ij}^{(m)}c_{ij}, m=\overline{0,8},$$

	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
(0)	1	1	ε	ε^2	ε^2	ε
(1)	1	1	1	1	1	1
(2)	1	1	ε^2	ε	ε	ε^2
(3)	1	1	1	1	1	1
(4)	1	-1	-1	-1	2	1
(5)	1	1	ε^2	ε	ε	ε^2
(6)	1	1	ε	ε^2	ε^2	ε
(7)	1	-1	$-\varepsilon$	$-\varepsilon^2$	ε^2	ε
(8)	1	-1	$-\varepsilon^2$	$-\varepsilon$	ε	ε^2

Таблица 1: $h_{d,m}$

где $t_{ij}^{(m)}$ коэффициент, стоящий у пары индексов (ij) в условной записи для (m). Через введённые таким образом новые переменные однозначно выражаются предыдущие.

Определим $\varphi_d(m)$ следующим образом. Возьмём произвольную запись (m). Применим ко всем парам индексов части перестановку φ_d. Полученный результат и будем обозначать, как $\varphi_d(m)$. Нетрудно показать, что $\varphi_d(m) = h_{d,m}\psi_d(m)$, $m = \overline{0,8}$, $d = \overline{1,6}$, где $h_{d,m}$ — константы, приведённые в таблице ниже, ψ_d — некоторые перестановки на множестве $\{0, 1, \ldots, 8\}$. Обозначим $\varphi_d(x_m) = h_{d,m}x_{\psi_d(m)}$; аналогично для переменных y, z, a, b, c.

3.3 Построение новой системы

Начнём строить систему в новых переменных. Для произвольной тройки (m, p, q) определим правую часть уравнения, как

$$2 \cdot (x_m y_p z_q + x_p y_q z_m + x_q y_m z_p)+$$
$$+2 \cdot (\varphi_2(x_m)\varphi_2(y_p)\varphi_2(z_q) + \varphi_2(x_p)\varphi_2(y_q)\varphi_2(z_m) + \varphi_2(x_q)\varphi_2(y_m)\varphi_2(z_p))+$$

$$+\sum_{d=3}^{6}(\varphi_d(a_m)\varphi_d(b_p)\varphi_d(c_q) + \varphi_d(a_p)\varphi_d(b_q)\varphi_d(c_m) + \varphi_d(a_q)\varphi_d(b_m)\varphi_d(c_p)). \quad (13)$$

При этом уравнения для для троек (m, p, q), (p, q, m), (q, m, p), $(\varphi_2(m), \varphi_2(p), \varphi_2(q))$, $(\varphi_2(p), \varphi_2(q), \varphi_2(m))$, $(\varphi_2(q), \varphi_2(m), \varphi_2(p))$ оказываются идентичными.

Из определения новых переменных следует, что

$$
\begin{gathered}
x_m y_p z_q + x_p y_q z_m + x_q y_m z_p = \\
= (\sum_{i=0}^{2}\sum_{j=0}^{2} t_{ij}^{(m)} x_{ij})(\sum_{k=0}^{2}\sum_{l=0}^{2} t_{kl}^{(p)} y_{kl})(\sum_{r=0}^{2}\sum_{s=0}^{2} t_{rs}^{(q)} z_{rs}) + \\
+(\sum_{k=0}^{2}\sum_{l=0}^{2} t_{kl}^{(p)} x_{kl})(\sum_{r=0}^{2}\sum_{s=0}^{2} t_{rs}^{(q)} y_{rs})(\sum_{i=0}^{2}\sum_{j=0}^{2} t_{ij}^{(m)} z_{ij}) + \\
+(\sum_{r=0}^{2}\sum_{s=0}^{2} t_{rs}^{(q)} x_{rs})(\sum_{i=0}^{2}\sum_{j=0}^{2} t_{ij}^{(m)} y_{ij})(\sum_{k=0}^{2}\sum_{l=0}^{2} t_{kl}^{(p)} z_{kl}) = \\
= \sum_{i=0}^{2}\sum_{j=0}^{2}\sum_{k=0}^{2}\sum_{l=0}^{2}\sum_{r=0}^{2}\sum_{s=0}^{2} t_{ij}^{(m)} t_{kl}^{(p)} t_{rs}^{(q)} (x_{ij} y_{kl} z_{rs} + x_{kl} y_{rs} z_{ij} + x_{rs} y_{ij} z_{kl}).
\end{gathered}
$$

Аналогично

$$
\begin{gathered}
\varphi_2(x_m)\varphi_2(y_p)\varphi_2(z_q) + \varphi_2(x_p)\varphi_2(y_q)\varphi_2(z_m) + \varphi_2(x_q)\varphi_2(y_m)\varphi_2(z_p) = \\
= \sum_{i,j,k,l,r,s=0}^{2} t_{ij}^{(m)} t_{kl}^{(p)} t_{rs}^{(q)} (x_{\varphi_2(ij)} y_{\varphi_2(kl)} z_{\varphi_2(rs)} + x_{\varphi_2(kl)} y_{\varphi_2(rs)} z_{\varphi_2(ij)} + x_{\varphi_2(rs)} y_{\varphi_2(ij)} z_{\varphi_2(kl)})
\end{gathered}
$$

и

$$
\begin{gathered}
\varphi_d(a_m)\varphi_d(b_p)\varphi_d(c_q) + \varphi_d(a_p)\varphi_d(b_q)\varphi_d(c_m) + \varphi_d(a_q)\varphi_d(b_m)\varphi_d(c_p) = \\
= \sum_{i,j,k,l,r,s=0}^{2} t_{ij}^{(m)} t_{kl}^{(p)} t_{rs}^{(q)} (a_{\varphi_d(ij)} b_{\varphi_d(kl)} c_{\varphi_d(rs)} + a_{\varphi_d(kl)} b_{\varphi_d(rs)} c_{\varphi_d(ij)} + a_{\varphi_d(rs)} b_{\varphi_d(ij)} c_{\varphi_d(kl)}).
\end{gathered}
$$

Таким образом, левую часть уравнения для тройки (m, p, q) можно записать, как

$$
\begin{gathered}
= \sum_{i,j,k,l,r,s=0}^{2} t_{ij}^{(m)} \cdot t_{kl}^{(p)} \cdot t_{rs}^{(q)} \cdot (2 \cdot (x_{ij} y_{kl} z_{rs} + x_{kl} y_{rs} z_{ij} + x_{rs} y_{ij} z_{kl}) + \\
+2 \cdot (x_{\varphi_2(ij)} y_{\varphi_2(kl)} z_{\varphi_2(rs)} + x_{\varphi_2(kl)} y_{\varphi_2(rs)} z_{\varphi_2(ij)} + x_{\varphi_2(rs)} y_{\varphi_2(ij)} z_{\varphi_2(kl)}) + \\
+ \sum_{d=3}^{6} a_{\varphi_d(ij)} b_{\varphi_d(kl)} c_{\varphi_d(rs)} + a_{\varphi_d(kl)} b_{\varphi_d(rs)} c_{\varphi_d(ij)} + a_{\varphi_d(rs)} b_{\varphi_d(ij)} c_{\varphi_d(kl)}).
\end{gathered}
$$

Следовательно, левая часть нового уравнения представляет собой линейную комбинацию левых частей уравнений из системы (12), причём уравнение для (ij, kl, rs) берётся с коэффициентом:

$$t_{ij}^{(m)} t_{kl}^{(p)} t_{rs}^{(q)}.$$

Обозначим правую часть уравнения для (ij, kl, rs) в системе (12) через $f_{ij,kl,rs}$. Тогда для того, чтобы уравнение для тройки (m, p, q) было линейной комбинацией уравнений из (12), его правая часть должна выглядеть, как

$$\sum_{i,j,k,l,r,s=0}^{2} t_{ij}^{(m)} t_{kl}^{(p)} t_{rs}^{(q)} f_{ij,kl,rs}. \tag{14}$$

Таким образом, используя ряд предположений, мы перешли от системы (7) к новой системе. Для каждой тройки (m, p, q) левая и правая части уравнения строятся по формулам (13) и (14), соответственно.

3.4 Описание программы

Программа для составления уравнений новой системы была написана на языке программирования Python с использованием пакета для символьных вычислений SymPy. Были найдены и загружены константы $h_{d,m}$, $m = \overline{0,8}$, $d = \overline{1,6}$, $t_{ij}^{(m)}$, $m = \overline{0,8}$, $i, j = \overline{0,2}$ и перестановки ψ_m, $m = \overline{0,8}$. Функции `generate_left` и `generate_right` вычисляют левую и правую части уравнения для каждой тройки (m, p, q). С помощью трёхмерного массива `is_counted` отсекались одинаковые уравнения, чтобы ускорить работу программы. Также программа вычисляет уравнения в "сжатом" виде.

4 Полученные результаты

Программа вычислила все уравнения, пропуская тройки, уравнения для которых уже были посчитаны. Также были исключены тождества. В итоге вышло 125 уравнений. Все коэффициенты получились целыми. Результат был сравнён с полученными ранее — все уравнения совпали.

Список литературы

[1] Strassen V. Gaussian elimination is not optimal // Numer. Math. 1969, том 13. С. 354–356.

5 Приложение

5.1 Исходный код

```python
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# загружаем зависимости
get_ipython().system('pip install sympy')
get_ipython().system('pip install numpy')


# In[2]:


# библиотека для символьных вычислений
from sympy import Rational, sqrt, I, symbols, simplify, Eq, latex
import numpy as np
from itertools import product


# In[3]:


# задаём константы

eps = -Rational(1, 2) + sqrt(3)/2*I

# t_{ij}^{(m)} <=> t[m][i][j]
t = np.array([
    [
        [eps, 0, 0],
        [0, 1, 0],
        [0, 0, eps**2],
```

```
    ],
    [
        [1, 0, 0],
        [0, 1, 0],
        [0, 0, 1],
    ],
    [
        [eps**2, 0, 0],
        [0, 1, 0],
        [0, 0, eps],
    ],
    [
        [0, 1, 1],
        [1, 0, 1],
        [1, 1, 0],
    ],
    [
        [0, -1, 1],
        [1, 0, -1],
        [-1, 1, 0],
    ],
    [
        [0, eps, 1],
        [eps, 0, eps**2],
        [1, eps**2, 0],
    ],
    [
        [0, eps**2, 1],
        [eps**2, 0, eps],
        [1, eps, 0],
    ],
    [
        [0, -eps**2, 1],
        [eps**2, 0, -eps],
        [-1, eps, 0],
    ],
    [
        [0, -eps, 1],
```

```
        [eps, 0, -eps**2],
        [-1, eps**2, 0],
    ],
])

t[4] *= I / sqrt(3)
t[7] *= I / sqrt(3)
t[8] *= I / sqrt(3)

# h_{d,m}
h = np.array([
    [1, 1, 1, 1, 1, 1, 1, 1, 1],  # phi1
    [1, 1, 1, 1, -1, 1, 1, -1, -1],  # phi2
    [eps, 1, eps**2, 1, -1, eps**2, eps, -eps, -eps**2],  # phi3
    [eps**2, 1, eps, 1, -1, eps, eps**2, -eps**2, -eps],  # phi4
    [eps**2, 1, eps, 1, 1, eps, eps**2, eps**2, eps],  # phi5
    [eps, 1, eps**2, 1, 1, eps**2, eps, eps, eps**2],  # phi6
])


# In[4]:


# задаём перестановки

# psi: (0, 1, 2, 3, 4, 5, 6, 7, 8)
def psi_even(m):
    return m

# psi: (2, 1, 0, 3, 4, 6, 5, 8, 7)


def psi_odd(m):
    match m:
        case 0:
            return 2
        case 2:
            return 0
```

```python
        case 5:
            return 6
        case 6:
            return 5
        case 7:
            return 8
        case 8:
            return 7
        case _:
            return m


psi = np.array([
    psi_even,
    psi_odd,
    psi_odd,
    psi_odd,
    psi_even,
    psi_even,
])


# In[5]:


# 54 переменные искомой системы
X = symbols("x:9")
Y = symbols("y:9")
Z = symbols("z:9")
A = symbols("a:9")
B = symbols("b:9")
C = symbols("c:9")


# In[6]:


# phi_d(x_m)
```

```
def phi(d, X, m):
    return h[d-1][m]*X[psi[d-1](m)]


# правая часть уравнения из исходной системы
def f(i, j, k, l, r, s):
    if (i == j and k == l and r == s) and not (i == j == k == l == r == s):
        return 1
    if (j == k and l == r and s == i) and not (i == j == k == l == r == s):
        return -1
    return 0


# In[7]:


# левая часть уравнения
def generate_left(m, p, q):
    left = 2*(X[m]*Y[p]*Z[q] + X[p]*Y[q]*Z[m] + X[q]*Y[m]*Z[p] +
              phi(2, X, m)*phi(2, Y, p)*phi(2, Z, q) +
              phi(2, X, p)*phi(2, Y, q)*phi(2, Z, m) +
              phi(2, X, q)*phi(2, Y, m)*phi(2, Z, p))
    for d in range(3, 7):
        left += phi(d, A, m)*phi(d, B, p)*phi(d, C, q) +            phi(d, A, p)*phi(d, B, q)*phi(d,
    return simplify(left)


# правая часть уравнения
def generate_right(m, p, q):
    right = 0
    for i, j, k, l, r, s in product(range(3), repeat=6):
        right += t[m][i][j]*t[p][k][l]*t[q][r][s]*f(i, j, k, l, r, s)
    return simplify(right)


def generate_equation(m, p, q):
    left = generate_left(m, p, q)
    right = generate_right(m, p, q)
```

```python
    return Eq(left, right)


equations = []
is_counted = np.zeros((9, 9, 9), bool)
with open("equations.tex", "w") as equations_file:
    for m, p, q in product(range(9), repeat=3):
        if is_counted[m][p][q]:
            continue

        # поскольку уравнения для шестёрок (m, p, q), (p, q, m), (q, m, p),
        # (phi_2(m), phi_2(p), phi_2(q)), (phi_2(p), phi_2(q), phi_2(m)), (phi_2(q), phi_2(m), phi_2(
        # оказываются идентичными
        is_counted[m][p][q] = True
        is_counted[p][q][m] = True
        is_counted[q][m][p] = True
        is_counted[psi[1](m)][psi[1](p)][psi[1](q)] = True
        is_counted[psi[1](p)][psi[1](q)][psi[1](m)] = True
        is_counted[psi[1](q)][psi[1](m)][psi[1](p)] = True

        equation = generate_equation(m, p, q)
        if equation == True:
            continue
        equations.append(equation)
        equations_file.write(f"\\[ {latex(equation)} \\]\n")


# In[8]:


# для уравнений в сжатом виде
XYZ = symbols("xyz:9:9:9")
ABC = symbols("abc:9:9:9")


# In[9]:


```

```
def get_index(m, p, q):
    return 81*m + 9*q + p


# phi_d(p_{mpq})
def phi(d, XYZ, m, p, q):
    return h[d-1][m]*h[d-1][p]*h[d-1][q]*XYZ[get_index(psi[d-1](m), psi[d-1](p), psi[d-1](q))]


# In[10]:


# левая часть уравнения
def generate_left(m, p, q):
    left = 2*(XYZ[get_index(m, p, q)] + phi(2, XYZ, m, p, q))
    for d in range(3, 7):
        left += phi(d, ABC, m, p, q)
    return simplify(left)


compressed_equations = []
is_counted = np.zeros((9, 9, 9), bool)
with open("compressed_equations.tex", "w") as equations_file:
    for m, p, q in product(range(9), repeat=3):
        if is_counted[m][p][q]:
            continue

        # поскольку уравнения для шестёрок (m, p, q), (p, q, m), (q, m, p),
        # (phi_2(m), phi_2(p), phi_2(q)), (phi_2(p), phi_2(q), phi_2(m)), (phi_2(q), phi_2(m), phi_2(
        # оказываются идентичными
        is_counted[m][p][q] = True
        is_counted[p][q][m] = True
        is_counted[q][m][p] = True
        is_counted[psi[1](m)][psi[1](p)][psi[1](q)] = True
        is_counted[psi[1](p)][psi[1](q)][psi[1](m)] = True
        is_counted[psi[1](q)][psi[1](m)][psi[1](p)] = True

        equation = generate_equation(m, p, q)
```

```
        if equation == True:
            continue
        compressed_equations.append(equation)
        equations_file.write(f"\\[ {latex(equation)} \\]\n")

```