



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н. Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н. Э. Баумана)

---

ФАКУЛЬТЕТ «Информатика и системы управления»

---

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

---

## ОТЧЕТ

по лабораторной работе № 2  
по курсу «Анализ алгоритмов»  
на тему: «Умножение матриц»

Студент ИУ7-54Б  
(Группа)

\_\_\_\_\_  
(Подпись, дата)

Булдаков М.  
(И. О. Фамилия)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

Волкова Л. Л.  
(И. О. Фамилия)

2023 г.

# СОДЕРЖАНИЕ

<b>ВВЕДЕНИЕ</b>	<b>3</b>
<b>1 Аналитический раздел</b>	<b>4</b>
1.1 Классический алгоритм . . . . .	4
1.2 Алгоритм Винограда . . . . .	4
1.3 Алгоритм Штрассена . . . . .	6
<b>2 Конструкторский раздел</b>	<b>8</b>
2.1 Требования к программному обеспечению . . . . .	8
<b>3 Технологический раздел</b>	<b>11</b>
<b>4 Исследовательский раздел</b>	<b>12</b>
<b>ЗАКЛЮЧЕНИЕ</b>	<b>13</b>
<b>СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ</b>	<b>14</b>

# ВВЕДЕНИЕ

Матрицы являются одним из основных инструментов линейной алгебры, они позволяют описывать и анализировать линейные отношения между различными объектами и явлениями. В настоящее время матрицы широко используются в науке, технике, экономике и других сферах человеческой деятельности.

Размеры матриц могут быть очень большими в зависимости от конкретной задачи, поэтому оптимизация алгоритмов обработки матриц является важной задачей программирования. Основной акцент будет сделан на оптимизации алгоритма умножения матриц.

Цель данной лабораторной работы – описать, реализовать и исследовать алгоритмы умножения матриц и их оптимизации. Для достижения поставленной цели необходимо выполнить следующие задачи.

- 1) Описать алгоритмы умножения матриц:
  - классический алгоритм;
  - алгоритм Винограда;
  - алгоритм Штрассена.
- 2) Оптимизировать перечисленные алгоритмы.
- 3) Разработать программное обеспечение, реализующее алгоритмы умножения.
- 4) Выбрать инструменты для реализации и замера процессорного времени выполнения алгоритмов.
- 5) Проанализировать затраты реализаций алгоритмов по времени и по памяти.

# 1 Аналитический раздел

Матрицей называется прямоугольная таблица чисел, вида (1.1), состоящая из  $m$  строк и  $n$  столбцов [1].

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}, \quad (1.1)$$

Пусть  $A$  – матрица, тогда  $a_{ij}$  – элемент этой матрицы, который находится на  $i$ -ой строке и  $j$ -ом столбце.

Если количество столбцов первой матрицы совпадает с количеством строк второй матрицы, то возможно выполнить их матричное умножение. В результате умножения получится матрица-произведение, количество строк в которой равно количеству строк первой матрицы, а количество столбцов равно количеству столбцов второй матрицы.

## 1.1 Классический алгоритм

Пусть даны две прямоугольные матрицы  $A$  и  $B$  размеров  $[m \times n]$  и  $[n \times k]$  соответственно. В результате произведения матриц  $A$  и  $B$  получим матрицу  $C$  размера  $[m \times k]$ , элементы которой вычисляются по (1.2).

$$c_{ij} = \sum_{l=1}^n a_{il}b_{lj} \quad (1.2)$$

Классический алгоритм умножения матриц, реализует формулу (1.2).

## 1.2 Алгоритм Винограда

Алгоритм Винограда – алгоритм умножения квадратных матриц с асимптотической сложностью  $O(n^{2,373})$  [2].

Рассмотрим два вектора  $V = (v_1, v_2, v_3, v_4)$  и  $W = (w_1, w_2, w_3, w_4)$ .

Их скалярное произведение равно (1.3).

$$V \cdot W = v_1 \cdot w_1 + v_2 \cdot w_2 + v_3 \cdot w_3 + v_4 \cdot w_4 \quad (1.3)$$

Равенство (1.3) можно переписать в виде (1.4)

$$V \cdot W = (v_1 + w_2) \cdot (v_2 + w_1) + (v_3 + w_4) \cdot (v_4 + w_3) - v_1 \cdot v_2 - v_3 \cdot v_4 - w_1 \cdot w_2 - w_3 \cdot w_4 \quad (1.4)$$

Теперь допустим, что у нас есть две матрицы  $A$  и  $B$  размерности  $m \times n$  и  $n \times p$  соответственно, и мы хотим найти их произведение  $C = A \cdot B$ . Тогда алгоритм будет состоять из следующих шагов:

- 1) Подготовительные вычисления. Сначала создаются два вспомогательных массива (1.5) и (1.6).

$$\text{rowFactor}[i] = \sum_{j=0}^{n/2-1} A[i][2j+1] \cdot A[i][2j] \quad (1.5)$$

для  $0 \leq i < m$

$$\text{colFactor}[j] = \sum_{i=0}^{n/2-1} B[2i+1][j] \cdot B[2i][j] \quad (1.6)$$

для  $0 \leq j < p$ .

- 2) Умножение матриц. Вычисляем результирующую матрицу  $C$  по формуле (1.7).

$$C[i][j] = \sum_{k=0}^{n/2-1} (A[i][2k+1] + B[2k][j]) \cdot (A[i][2k] + B[2k+1][j]) - \text{rowFactor}[i] - \text{colFactor}[j] \quad (1.7)$$

для  $0 \leq i < m$  и  $0 \leq j < p$ .

- 3) Коррекция. Если  $n$  нечетно, добавляем коррекцию, в соответствии с формулой (1.8).

$$C[i][j] += A[i][n] \cdot B[n][j] \quad (1.8)$$

для  $0 \leq i < m$  и  $0 \leq j < p$ .

### 1.3 Алгоритм Штрассена

Алгоритм Штрассена — это алгоритм умножения квадратных матриц, который является более эффективным для больших матриц, чем классический метод умножения [3].

Если добавить к матрицам  $A$  и  $B$  одинаковые нулевые строки и столбцы, их произведение станет равно матрице  $C$  с теми же добавленными строками и столбцами. Поэтому в данном алгоритме рассматриваются матрицы порядка  $m \cdot 2^{k+1}$ , где  $k \in \mathbb{N}$ , а все остальные матрицы, сводятся к этому размеру добавлением нулевых строк и столбцов.

Алгоритм состоит из следующих шагов:

- 1) Разбиение матриц (1.9).

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad (1.9)$$

, где  $A_{ij}$  и  $B_{ij}$  — матрицы порядка  $m \cdot 2^k$

- 2) Вычисление вспомогательных матриц (1.10).

$$\begin{aligned} M_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\ M_2 &= (A_{21} + A_{22})B_{11} \\ M_3 &= A_{11}(B_{12} - B_{22}) \\ M_4 &= A_{22}(B_{21} - B_{11}) \\ M_5 &= (A_{11} + A_{12})B_{22} \\ M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ M_7 &= (A_{12} - A_{22})(B_{21} + B_{22}) \end{aligned} \quad (1.10)$$

- 3) Вычисление результирующих подматриц (1.11).

$$\begin{aligned} C_{11} &= M_1 + M_4 - M_5 + M_7 \\ C_{12} &= M_3 + M_5 \\ C_{21} &= M_2 + M_4 \\ C_{22} &= M_1 - M_2 + M_3 + M_6 \end{aligned} \quad (1.11)$$

Результирующая матрица состоит из  $C_{ij}$  (1.12).

$$AB = C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \quad (1.12)$$

## Вывод

В данном разделе были описаны алгоритмы классической матричной мультипликации, алгоритм Штрассена и алгоритм Винограда.

## 2 Конструкторский раздел

В этом разделе будет представлено описание используемых типов данных, а также схематические изображения алгоритмов матричного умножения - стандартного, Штрассена, алгоритма Винограда и оптимизированного алгоритма Винограда.

### 2.1 Требования к программному обеспечению

Программа должна поддерживать два режима работы: режим массового замера времени и режим умножения матриц.

Режим массового замера времени должен обладать следующим функционалом:

- генерировать матрицы различного размера для проведения замеров;
- осуществлять массовый замер, используя сгенерированные данные;
- результаты массового замера должны быть представлены в виде таблицы и графика.

К режиму умножения матриц выдвигается ряд требований:

- возможность работать с матрицами разного размера, которые вводит пользователь;
- наличие интерфейса для выбора действий;
- проверять возможность умножения матриц.



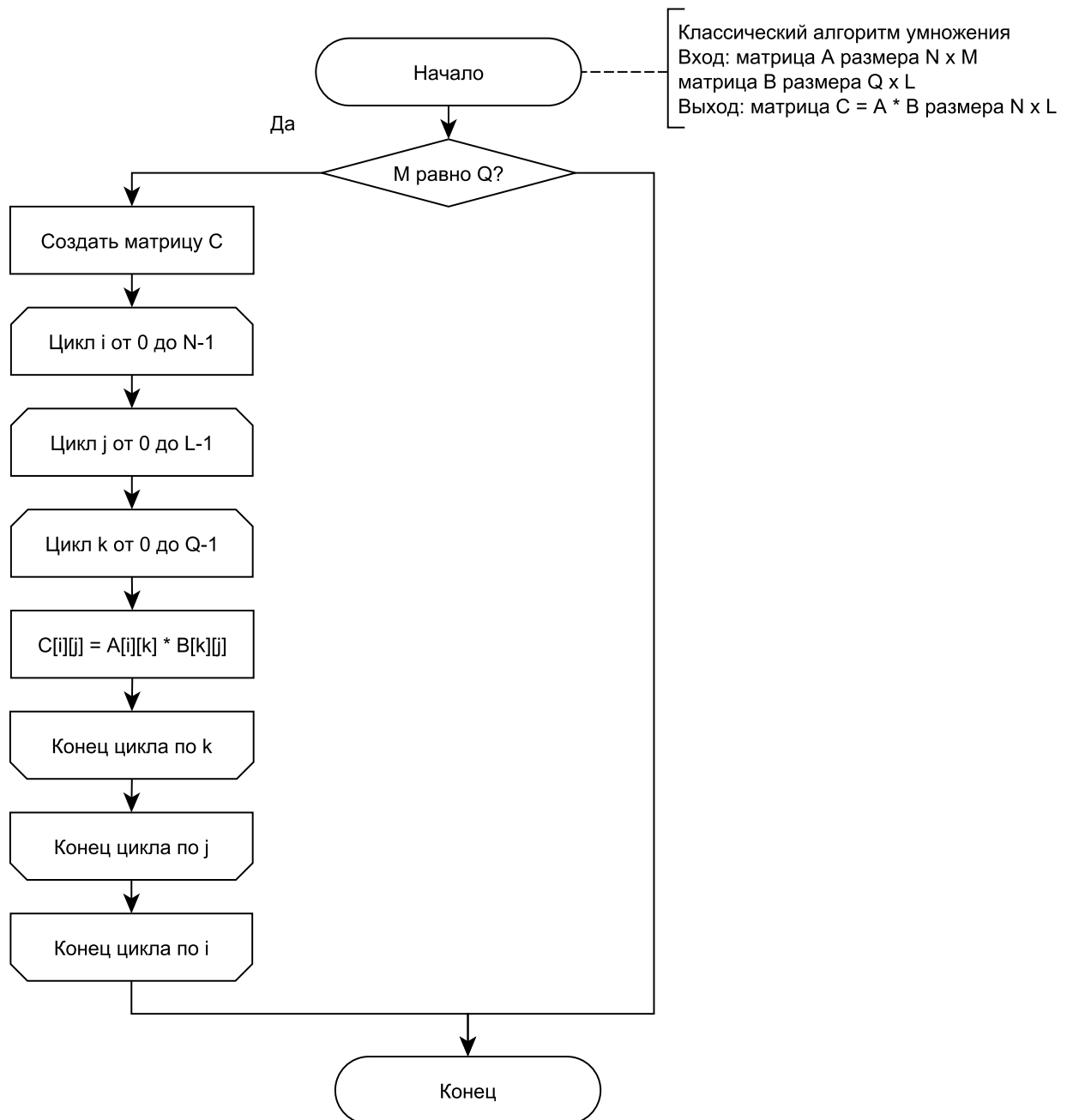


Рисунок 2.1 – Классический алгоритм умножения матриц

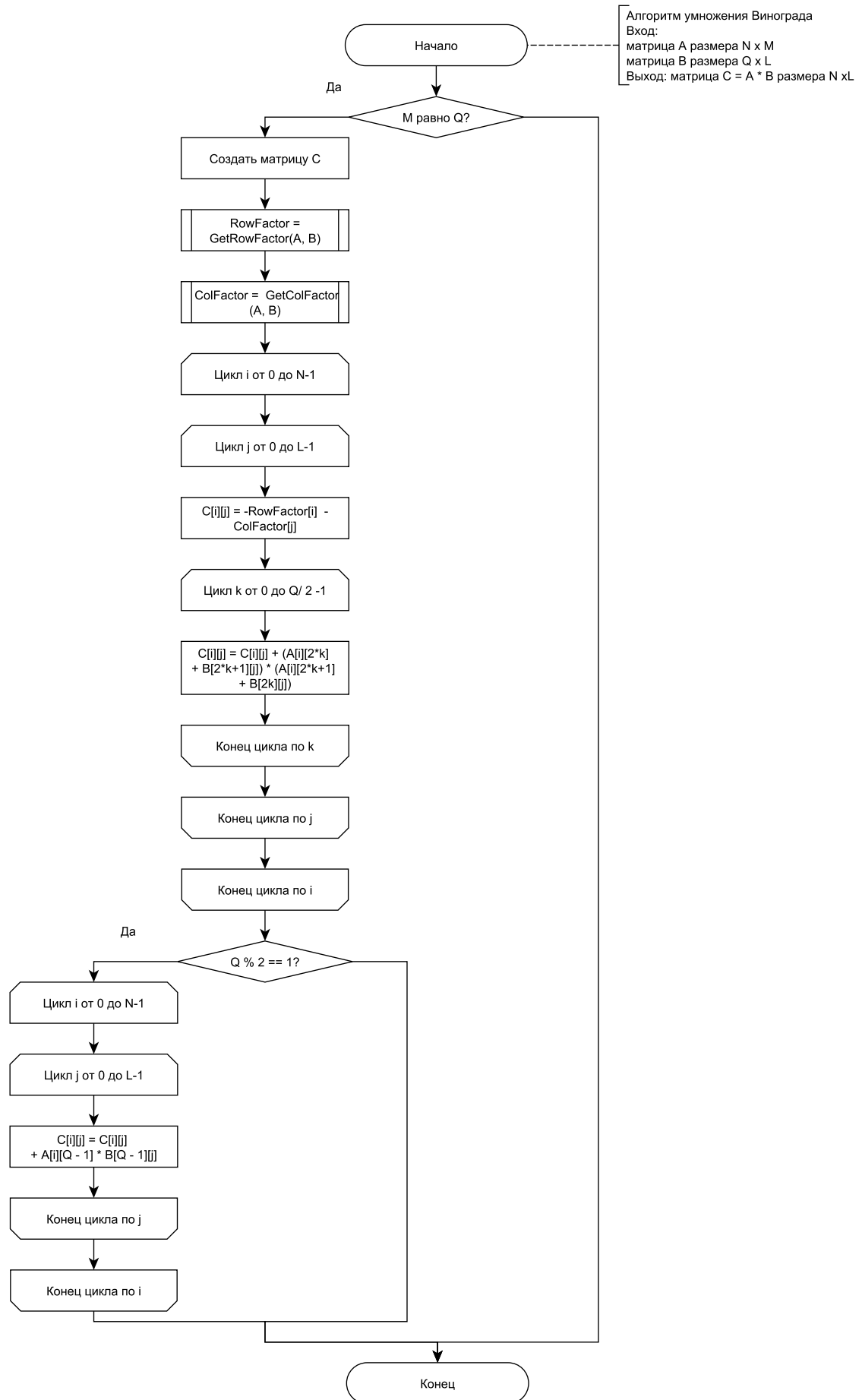


Рисунок 2.2 – Алгоритм умножения матриц Винограда

### 3 Технологический раздел

## 4 Исследовательский раздел

## ЗАКЛЮЧЕНИЕ

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Конспект лекций, читаемых в курсе «Высшая математика» Южного федерального университета. — Режим доступа: [http://mmtb.uginfo.sfedu.ru/algebra/Print/print\\_I-1.pdf](http://mmtb.uginfo.sfedu.ru/algebra/Print/print_I-1.pdf) (дата обращения: 12.10.2023).
2. Головашкин Д. Л. Векторные алгоритмы вычислительной линейной алгебры: учеб. пособие. // — — Самара: Изд-во Самарского университета, 2019. — С. 28—35.
3. Gaussian Elimination is not Optimal. — Режим доступа: [https://www2.math.uconn.edu/~leykekhman/courses/MATH\\_5510/fa\\_2016/papers/fast\\_matrix.pdf](https://www2.math.uconn.edu/~leykekhman/courses/MATH_5510/fa_2016/papers/fast_matrix.pdf) (дата обращения: 13.10.2023).