



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»

«Трудоёмкость алгоритмов умножения матриц»

Студент \_\_\_\_\_ Маслова Марина Дмитриевна

Группа \_\_\_\_\_ ИУ7-53Б

Оценка (баллы) \_\_\_\_\_

Преподаватель \_\_\_\_\_ Волкова Лилия Леонидовна

2021 г.

# Содержание

<b>Введение</b>	<b>3</b>
<b>1 Аналитическая часть</b>	<b>4</b>
1.1 Стандартный алгоритм умножения матриц . . . . .	4
1.2 Алгоритм умножения матриц Винограда . . . . .	4
1.3 Вывод . . . . .	5
<b>2 Конструкторская часть</b>	<b>6</b>
2.1 Разработка алгоритмов . . . . .	6
2.2 Структура разрабатываемого ПО . . . . .	6
2.3 Классы эквивалентности при тестировании . . . . .	6
2.4 Вывод . . . . .	6
<b>3 Технологическая часть</b>	<b>7</b>
3.1 Требования к программному обеспечению . . . . .	7
3.2 Средства реализации . . . . .	7
3.3 Листинги кода . . . . .	7
3.4 Описание тестирования . . . . .	8
3.5 Вывод . . . . .	9
<b>4 Исследовательская часть</b>	<b>10</b>
4.1 Технические характеристики . . . . .	10
4.2 Примеры работы программы . . . . .	10
4.3 Результаты тестирования . . . . .	10
4.4 Постановка эксперимента по замеру времени . . . . .	11
4.5 Результаты эксперимента . . . . .	11
4.6 Вывод . . . . .	11
<b>Заключение</b>	<b>12</b>
<b>Список литературы</b>	<b>13</b>

# Введение

Умножение матриц является основным инструментом линейной алгебры и имеет многочисленные применения в математике, физике, программировании [1]. При этом сложность стандартного алгоритма умножения матриц  $N \times N$  составляет  $O(N^3)$  [2], что послужило причиной разработки новых алгоритмов меньшей сложности. Одним из них является алгоритм Винограда с асимптотической сложностью  $O(N^{2.3755})$  [1].

**Целью данной работы** является изучение алгоритмов умножения матриц: стандартного и Винограда, — а также получение навыков расчета сложности алгоритмов и их оптимизации.

Для достижения поставленной цели необходимо выполнить следующие **задачи**:

- изучить алгоритмы умножения матриц: стандартный и алгоритм Винограда;
- разработать каждый из алгоритмов;
- дать теоретическую оценку трудоемкости стандартного алгоритма и алгоритма Винограда;
- оптимизировать алгоритм Винограда и дать теоретическую оценку его трудоемкости;
- реализовать каждый из трех алгоритмов;
- провести тестирование реализованных алгоритмов;
- провести сравнительный анализ алгоритмов по процессорному времени работы реализации.

# 1 Аналитическая часть

В данном разделе представлено теоретическое описание стандартного алгоритма умножения матриц и алгоритма Винограда.

## 1.1 Стандартный алгоритм умножения матриц

Стандартный алгоритм умножения матриц является реализацией математического определения произведения матриц, которое формулируется следующим образом:

пусть даны матрица  $A = (a_{ij})_{m \times n}$ , имеющая  $m$  строк и  $n$  столбцов, и матрица  $B = (b_{ij})_{n \times k}$ , имеющая  $n$  строк и  $k$  столбцов, тогда матрица  $C = (c_{ij})_{m \times p}$ , имеющая  $m$  строк и  $p$  столбцов, где:

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj} = \sum_{k=1}^n a_{ik}b_{kj}; \quad (1.1)$$
$$i = 1, 2, \dots, m; \quad j = 1, 2, \dots, p;$$

— называется **произведением** матриц  $A$  и  $B$  [3].

## 1.2 Алгоритм умножения матриц Винограда

Из формулы 1.1 видно, что каждый элемент итоговой матрицы представляет собой скалярное произведение соответствующих строки и столбца исходных матриц, которое, в свою очередь, допускает предварительную обработку, т. е. часть вычислений можно выполнить заранее.

Пусть даны два вектора  $V = (v_1, v_2, v_3, v_4)$  и  $W = (w_1, w_2, w_3, w_4)$ . Их скалярное произведение представлено формулой 1.2:

$$V \cdot W = v_1w_1 + v_2w_2 + v_3w_3 + v_4w_4 \quad (1.2)$$

Равенство 1.2 можно представить в виде:

$$V \cdot W = (v_1 + w_2)(v_2 + w_1) + (v_3 + w_4)(v_4 + w_3) - v_1v_2 - v_3v_4 - w_1w_2 - w_3w_4 \quad (1.3)$$

Хотя в выражении 1.3 больше операций, чем в выражении 1.2: девять

сложений против трех, и шесть умножений против четырех, — последнее выражение допускает предварительную обработку, а именно операции вида  $v_i v_{i+1} + v_{i+2} v_{i+3}$  и  $w_i w_{i+1} + w_{i+2} w_{i+3}$  для  $i \in \overline{0, 4..n}$  могут быть вычислены заранее. Это позволить нам для каждого элемента выполнять только операции для двух первых слагаемых в выражении 1.3, то есть два умножения и пять сложений, а также две операции сложения для учета уже вычисленных значений [4] .

### 1.3 Вывод

## **2 Конструкторская часть**

### **2.1 Разработка алгоритмов**

### **2.2 Структура разрабатываемого ПО**

Для реализации разрабатываемого программного обеспечения будет использоваться метод структурного программирования. Каждый из алгоритмов будет представлен отдельной функцией, при наличии части инициализации матрицы, она также будет вынесена в отдельную функцию. Также будут реализованы функции для ввода-вывода и функция, вызывающая все подпрограммы для связности и полноценности программы.

### **2.3 Классы эквивалентности при тестировании**

Для тестирования программного обеспечения во множестве тестов будут выделены следующие классы эквивалентности:

—  
—  
—  
—  
—  
—  
—  
—  
—

### **2.4 Вывод**

## 3 Технологическая часть

В данном разделе описаны требования к программному обеспечению, средства реализации, приведены листинги кода и данные, на которых будет проводиться тестирование.

### 3.1 Требования к программному обеспечению

Программа должна предоставлять следующие возможности:

- выбор режима работы: для единичного эксперимента и для массовых экспериментов;
- в режиме единичного эксперимента ввод двух строк на русском или английском языках и вывод полученных разными реализациями расстояний;
- в режиме массовых экспериментов измерение времени при различных длинах строк и построение графиков по полученным данным.

### 3.2 Средства реализации

Для реализации данной лабораторной работы выбран интерпретируемый язык программирования высокого уровня Python[5], так как он позволяет реализовывать сложные задачи за кратчайшие сроки за счет простоты синтаксиса и наличия большого количества подключаемых библиотек.

В качестве среды разработки выбран текстовый редактор Vim[6] с установленными плагинами автодополнения и поиска ошибок в процессе написания, так как он реализует быстрое перемещение по тексту программы и простое взаимодействие с командной строкой.

Замеры времени проводились при помощи функции `process_time_ns` из библиотеки `time`[7].

### 3.3 Листинги кода

В данном подразделе представлены листинги кода ранее описанных алгоритмов:

- (листинг ??);

- (листинг ??-??);
- (листинг ??, ??);
- (листинг ??).

### **3.4 Описание тестирования**

В таблице ?? приведены функциональные тесты для алгоритмов



### **3.5 Вывод**

## **4 Исследовательская часть**

### **4.1 Технические характеристики**

Технические характеристики устройства, на котором выполнялось тестирование:

- Операционная система: Manjaro [8] Linux x86\_64.
- Память: 8 GiB.
- Процессор: Intel® Core™ i5-8265U[9].

Тестирование проводилось на ноутбуке, включенном в сеть электропитания. Во время тестирования ноутбук был нагружен только встроенными приложениями окружения, окружением, а также непосредственно системой тестирования.

### **4.2 Примеры работы программы**

В данном подразделе представлены примеры работы программы. На рисунке ?? приведен пример работы программы при вводе строк в русской раскладке и равными расстояниями Левенштейна и Дамерау-Левенштейна. На рисунке ?? приведен пример работы программы при вводе строк в английской раскладке и разными полученными значениями расстояний.

### **4.3 Результаты тестирования**

В таблице ?? приведены результаты работы программы на тестах, описанных в таблице ?. В результате сравнения ожидаемого и полученного результата делаем вывод, что все тесты были пройдены.

#### **4.4 Постановка эксперимента по замеру времени**

#### **4.5 Результаты эксперимента**

#### **4.6 Вывод**

# Заключение

В ходе выполнения лабораторной работы:

—  
—  
—  
—  
—

# Список литературы

- [1] Анисимов Н. С. Строганов Ю. В. Реализация алгоритма умножения матриц по Винограду на языке Haskell // Новые информационные технологии в автоматизированных системах. 2018. № 21. С. 390–395. Режим доступа: <https://cyberleninka.ru/article/n/realizatsiya-algoritma-umnozheniya-matrits-po-vinogradu-na-yazyke-haskell> (дата обращения: 30.10.2021).
- [2] Späth Till. Searching for fast matrix multiplication algorithms. Ph.D. thesis. 2019. 10.
- [3] Шихобалов Л. С. Матрицы и определители : учеб. пособие. СПб.: СПбГУ, 2015. с. 55.
- [4] Макконелл Дж. Основы современных алгоритмов. 2-е доп. изд. М.: Техносфера, 2004. с. 368.
- [5] Welcome to Python [Электронный ресурс]. Режим доступа: <https://www.python.org> (дата обращения: 12.10.2021).
- [6] welcome home : vim online [Электронный ресурс]. Режим доступа: <https://www.vim.org/> (дата обращения: 12.10.2021).
- [7] time — Time access and conversions [Электронный ресурс]. Режим доступа: [https://docs.python.org/3/library/time.html#time.process\\_time\\_ns](https://docs.python.org/3/library/time.html#time.process_time_ns) (дата обращения: 04.10.2021).
- [8] Manjaro — enjoy the simplicity [Электронный ресурс]. Режим доступа: <https://manjaro.org/> (дата обращения: 17.10.2021).
- [9] Процессор Intel® Core™ i5-8265U [Электронный ресурс]. Режим доступа: <https://ark.intel.com/content/www/ru/ru/ark/products/149088/intel-core-i5-8265u-processor-6m-cache-up-to-3-90-ghz.html> (дата обращения: 17.10.2021).