МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3**

**по дисциплине  
 «АЛГОРИТМЫ И АНАЛИЗ СЛОЖНОСТИ»**

Выполнил студент группы 25/2                                       А.А. Козин

Направление подготовки  02.03.03  Математическое обеспечение и администрирование информационных систем

Курс    2

Отчет принял доктор физико-математических наук, профессор                                                                                       А.И. Миков

Краснодар

2021 г.

**Задание**: Найти функцию сложности для алгоритма Штрассена.

**Решение**.

Для анализа сложности алгоритма Штрассена мы приведём его в буквальном виде, поставив счётчик.

**Текст программы:**

#include <iostream>

#include <vector>

#include <ctime>

using namespace std;

int counter = 0;

int\*\* init(int n) {

int\*\* A = new int\* [n];

for (int i = 0; i < n; i++) {

A[i] = new int[n];

}

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

A[i][j] = 0;

}

}

return A;

}

int\*\* suma(int\*\* A, int\*\* B, int n) {

int\*\* C = init(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

C[i][j] = A[i][j] + B[i][j];

counter += 2;

}

}

return C;

}

int\*\* raznost(int\*\* A, int\*\* B, int n) {

int\*\* C = init(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

C[i][j] = A[i][j] - B[i][j];

counter += 2;

}

}

return C;

}

int\*\* proiz(int\*\* A, int\*\* B, int n) {

int\*\* C = init(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

for (int k = 0; k < n; k++) {

C[i][j] += A[i][k] \* B[k][j];

counter += 3;

}

}

}

return C;

}

int\*\* strassen(int\*\* A, int\*\* B, int n) {

if (n <= 32) return proiz(A, B, n);

int i, j;

int\*\* C = init(n);

int\*\* A11 = init(n / 2); int\*\* A12 = init(n / 2); int\*\* A21 = init(n / 2); int\*\* A22 = init(n / 2);

int\*\* B11 = init(n / 2); int\*\* B12 = init(n / 2); int\*\* B21 = init(n / 2); int\*\* B22 = init(n / 2);

int\*\* C11 = init(n / 2); int\*\* C12 = init(n / 2); int\*\* C21 = init(n / 2); int\*\* C22 = init(n / 2);

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

if (i < n / 2 && j < n / 2) {

A11[i][j] = A[i][j]; B11[i][j] = B[i][j];

}

if (i < n / 2 && j >= n / 2) {

A12[i][j - (n / 2)] = A[i][j]; B12[i][j - (n / 2)] = B[i][j];

}

if (i >= n / 2 && j < n / 2) {

A21[i - (n / 2)][j] = A[i - (n / 2)][j]; B21[i - (n / 2)][j] = B[i][j];

}

if (i >= n / 2 && j >= n / 2) {

A22[i - (n / 2)][j - (n / 2)] = A[i][j]; B22[i - (n / 2)][j - (n / 2)] = B[i][j];

}

}

}

int\*\* P1 = init(n / 2); int\*\* P2 = init(n / 2); int\*\* P3 = init(n / 2); int\*\* P4 = init(n / 2);

int\*\* P5 = init(n / 2); int\*\* P6 = init(n / 2); int\*\* P7 = init(n / 2);

P1 = strassen(suma(A11, A22, n / 2), suma(B11, B22, n / 2), n / 2);

P2 = strassen(suma(A21, A22, n / 2), B11, n / 2);

P3 = strassen(A11, raznost(B12, B22, n / 2), n / 2);

P4 = strassen(A22, raznost(B21, B11, n / 2), n / 2);

P5 = strassen(suma(A11, A12, n / 2), B22, n / 2);

P6 = strassen(raznost(A21, A11, n / 2), suma(B11, B12, n / 2), n / 2);

P7 = strassen(raznost(A12, A22, n / 2), suma(B21, B22, n / 2), n / 2);

counter += 7;

C11 = raznost(suma(P1, P4, n / 2), suma(P5, P7, n / 2), n / 2);

C12 = suma(P3, P5, n / 2);

C21 = suma(P2, P4, n / 2);

C22 = suma(raznost(P1, P2, n / 2), suma(P3, P6, n / 2), n / 2);

counter += 4;

for (i = 0; i < n / 2; i++) {

for (j = 0; j < n / 2; j++) {

C[i][j] = C11[i][j];

C[i][j + (n / 2)] = C12[i][j];

C[i + (n / 2)][j] = C21[i][j];

C[i + (n / 2)][j + (n / 2)] = C22[i][j];

}

}

return C;

}

void main() {

srand(time(0));

int n = 64;

cout << "n " <<" counter"<< endl;

cout << "--------------------------" << endl;

for (int k = 0; k < 5; k++) {

int\*\* A = init(n); int\*\* B = init(n); int\*\* C = init(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

A[i][j] = rand() % 10;

B[i][j] = rand() % 10;

C[i][j] = rand() % 10;

}

}

C = strassen(A, B, n);

cout << n <<" "<< counter << endl;

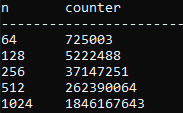
counter = 0;

n \*= 2;

}

}

**Программа выдает на консоль результаты в следующем виде:**



**График роста количества операций:**

**Анализ зависимости:**

Высчитывая формулу сложности алгоритма Штрассена применим теорему Мастера:

Т.к , где с - сложность каждого этапа рекурсии, а – количество рекурсивных подзадач в алгоритме, размерность исходной задачи каждый раз уменьшается в b раз, тогда, подставив значения, получим:

А значит, сложность алгоритма примерно равна .

**Вывод**: Сложность алгоритма Штрассена: