Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Новосибирский государственный технический университет»

Кафедра теоретической и прикладной информатики

Расчетно-графическая работа по дисциплине «Архитектура ЭВМ и ВС»

Факультет: ПМИ

Группа: ПМИ-03

Студент: Сидоров Даниил

Преподаватели: Маркова В.П, Щукин Г.А.

Новосибирск

2021

1. Цель работы

- Научиться измерять время работы программы.
- Научиться оценивать производительность ЭВМ на тестовых задачах.
- Научиться находить фрагменты программы, подлежащих оптимизации.

2. Условие задачи

Написать на языке С++ программу умножения двух квадратных матриц. Проверить правильность работы программы на нескольких тестовых наборах входных данных, минимальный размер матриц выбирается такой, при котором умножение происходит за 0,1- 0,5 секунд, а максимальный – при котором умножение происходит за 30-50 секунд.

Затем модифицировать программу: матрицу, обход которой происходит по столбцам, транспонировать до умножения. То есть при умножении элементы обоих матриц будут считываться построчно.

3. Текст программы

1) Программа умножения без транспонирования:

```
void input(int N, int **M) {
	int i, j;
	for (i = 0; i < N; i++) {
		for (j = 0; j < N; j++) {
			M[i][j] = rand()%100;
		}
	}
}
void AB(int N, int **A, int **B, int **C) {
	int i, j, k;
	for (i = 0; i < N; i++) {
		for (j = 0; j < N; j++) {
			C[i][j] = 0;
			for (k = 0; k < N; k++)
				C[i][j] += A[i][k] * B[k][j];
		}
	}
}
int main() {
	setlocale(LC_ALL, "rus");
	srand(time(0));
	int N;
	cout << "Размерность матрицы: ";
	cin >> N;
	int **A = new int *[N];
	int **B = new int *[N];
	int **C = new int *[N];
	for (int i = 0; i < N; i++) {
		A[i] = new int[N];
		B[i] = new int[N];
		C[i] = new int[N];
	}
```

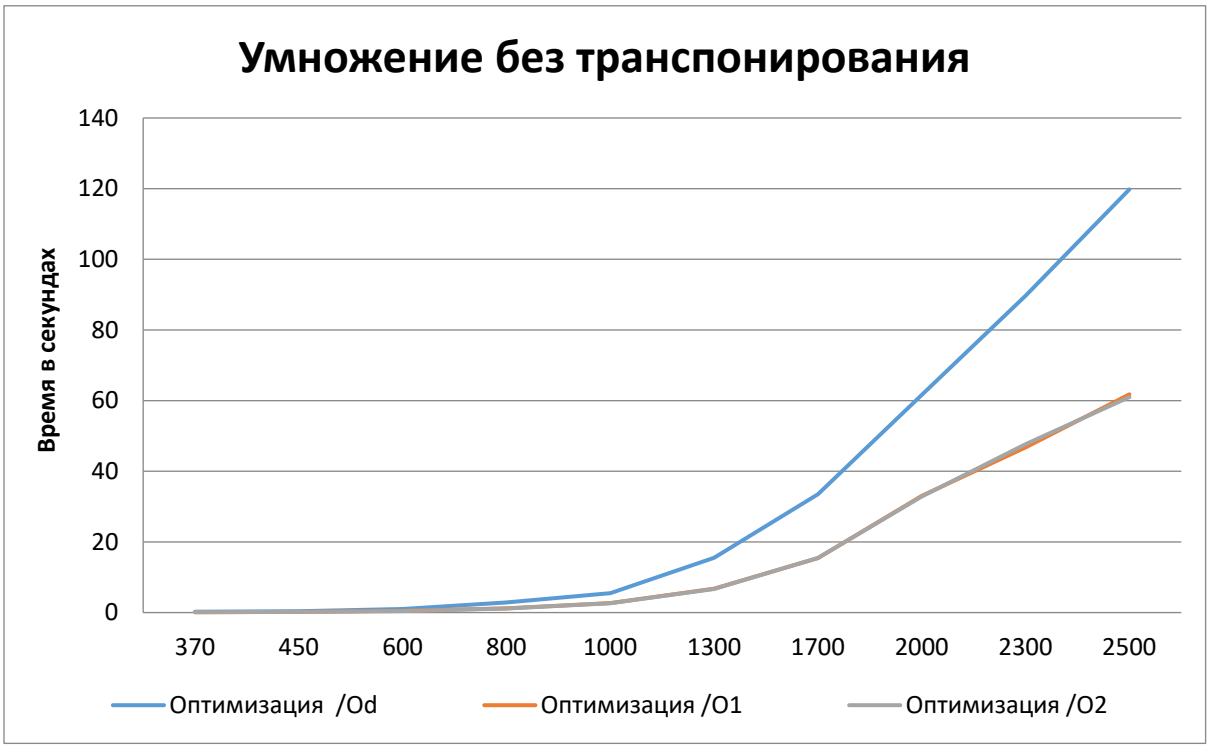
```
	input(N, A);
	input(N, B);
	unsigned int start = clock();
	AB(N, A, B, C);
	unsigned int end = clock();
	cout << endl << "Время выполнения подпрограммы " << (end-start) / 1000.0 <<
" сек";
	delete[] A;
	delete[] B;
	delete[] C;
	return 0;
```

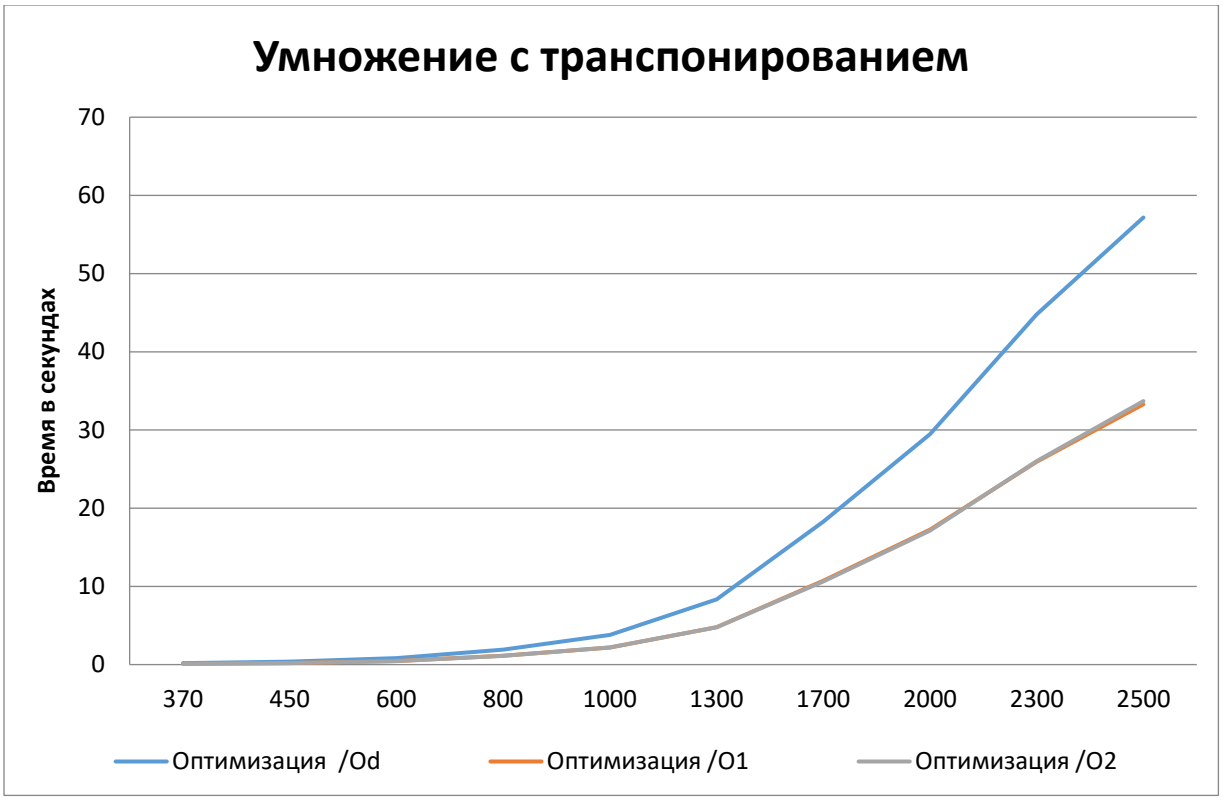
2)Программа умножения с транспонированием:

```
#include <iostream>
using namespace std;
void input(int N, int **M) {
	int i, j;
	for (i = 0; i < N; i++) {
		for (j = 0; j < N; j++) {
			M[i][j] = rand()%100;
		}
	}
}
void transpose(int N, int **B)
{
	int t;
	for (int i = 0; i < N; ++i)
	{
		for (int j = i; j < N; ++j)
		{
			t = B[i][j];
			B[i][j] = B[j][i];
			B[j][i] = t;
		}
	}
}
void ABT(int N, int **A, int **B, int **C) {
	int i, j, k;
	for (i = 0; i < N; i++) {
		for (j = 0; j < N; j++) {
			C[i][j] = 0;
			for (k = 0; k < N; k++)
				C[i][j] += A[i][k] * B[j][k];
		}
	}
}
int main() {
	setlocale(LC_ALL, "rus");
	srand(time(0));
	int N;
	cout << "Размерность матрицы: ";
	cin >> N;
	int **A = new int *[N];
	int **B = new int *[N];
	int **C = new int *[N];
	for (int i = 0; i < N; i++) {
		A[i] = new int[N];
```

```
            B[i] = new int[N];
            C[i] = new int[N];
        }
        input(N, A);
        input(N, B);
        transpose(N, B);
        unsigned int start = clock();
        ABT(N, A, B, C);
        unsigned int end = clock();
        cout << endl << "Время выполнения подпрограммы " << (end-start) / 1000.0 <<
" сек";
        delete[] A;
        delete[] B;
        delete[] C;
        return 0;
}
```

4. Графики зависимости времени счета от размера матриц:





5. Вывод

Для измерения времени работы программы использовались одинаковые наборы тестов, представленные на диаграмме.

1. Время работы программы при простом умножении матриц:
Без оптимизации 0,238- 119,795
С оптимизацией /O1 0,103-61,776 (уменьшилось в 1,94 раза)
С оптимизацией /O2 0,108-60,994 (уменьшилось в 1,96 раза)

2. Время работы программы при умножении матриц с транспонированной матрицей B:
Без оптимизации 0,184- 57,178
С оптимизацией /O1 0,103-33,294 (уменьшилось в 1,72 раза)
С оптимизацией /O2 0,110-33,697 (уменьшилось в 1,70 раза)

Таким образом, мы получаем, что перемножение с транспонированием матриц занимает примерно вдвое меньше времени, чем обычное перемножение матриц. Это становится заметно при больших размерностях матриц.

Работа выполнялась в Microsoft Visual Studio 2019 на языке C++
Технические характеристики ЭВМ:
Процессор: AMD Ryzen 5 3550H 2.10(Базовая)-3.70(Макс.) GHz
ОЗУ:8 Гб
Тип системы: 64 – разрядная операционная система, процессор x64