Государственное образовательное учреждение высшего профессионального образования
«Московский государственный технический университет имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

Лабораторная работа №...

по курсу «Анализ алгоритмов»

Тема лабораторной работы

Выполнил: Тимонин А.С., гр. ИУ7-52Б

Преподаватели: Волкова Л.Л., Строганов Ю.В.

2019 г.

Оглавление

Введение

1. Аналитический раздел

бла-бла-бла

1.1 Описание алгоритмов

бла-бла-бла

1.2 Алгоритм1

1.3 Алгоритм2

бла-бла-бла

2. Конструкторский раздел

bla-bla-bla

2.1 Разработка алгоритмов

тут схема алгоритмов

2.2 Расчет сложности

бла-бла-бла

2.3 Вывод

бла-бла-бла

- бла-бла-бла
- бла-бла-бла
- бла-бла-бла

3. Технологический раздел

бла-бла-бла

3.1 Требования к программному обеспечению

бла-бла-бла

3.2 Средства реализации

Для выполнения поставленной задачи был использован язык программирования C++. Среда для разработки XCode. Для измерения процессорного времени была взята функция rdtsc из библиотеки ctime.

Данный язык обусловлен тем, что функции замеры времени могут считывать не только абсолютное время, но и процессорное.

Версия компилятора C++: GNU++14 [-std=gnu++14]

3.3 Листинг кода

бла-бла-бла

Листинг 3.1: бла-бла-бла

```
1 void name(int a)
2 {
3   return;
4 }
```

3.4 Вывод

бла-бла-бла

4. Экспериментальный раздел

В данном разделе будет приведено экспериментальное исследование временных затрат разработанного программного обеспечения, вместе в подробным сравнительным анализом реализованных алгоритмов на основе экспериментальных данных.

4.1 Сравнительный анализ

Замеры времени выполнялись на квадратных матрицах размеров от 100x100 до 1000x1000 с интервалом в 100 элементов. Также замеры времени проводились над квадратными матрицами нечетной размерностью от 101x101 до 1001x1001 с шагом 100. В табл. 4.1-4.2 и рис. 4.1-4.1 представлены результаты замеров времени в тактах.

Замеры времени выполнялись на тестах..... Все замеры проводились на процессоре 1,4 GHz Intel Core i5 с памятью 8 ГБ 2133 MHz LPDDR3.

Таблица 4.1: Сравнение времени работы однопоточной и многопоточной версий алгоритма в единицах измерения библиотеки chrono

Размерность матриц	**Поток 1**	**Поток 2**	**Поток 4**	**Поток 8**
100	4	1	1	1
200	29	10	7	6
300	134	49	25	25
400	365	138	102	71
500	743	283	185	135
600	1206	450	247	210
700	1886	757	435	364
800	3029	1112	782	531
900	7909	2463	1028	860
1000	12799	3660	2047	1476

Рис. 4.1: График зависимости времени работы однопоточной и многопоточных версий алгоритма для четной размерности матриц

Таблица 4.2: Сравнение времени работы однопоточной и многопоточной версий алгоритма в единицах измерения библиотеки chrono

Размерность матриц	**Поток 1**	**Поток 2**	**Поток 4**	**Поток 8**
101	4	2	1	1
201	30	11	5	6
301	120	51	25	25
401	363	135	82	65
501	707	267	143	128
601	1202	444	236	206
701	1954	782	391	346
801	3161	1174	592	640
901	6120	2502	1305	1204
1001	9314	3766	2400	2186

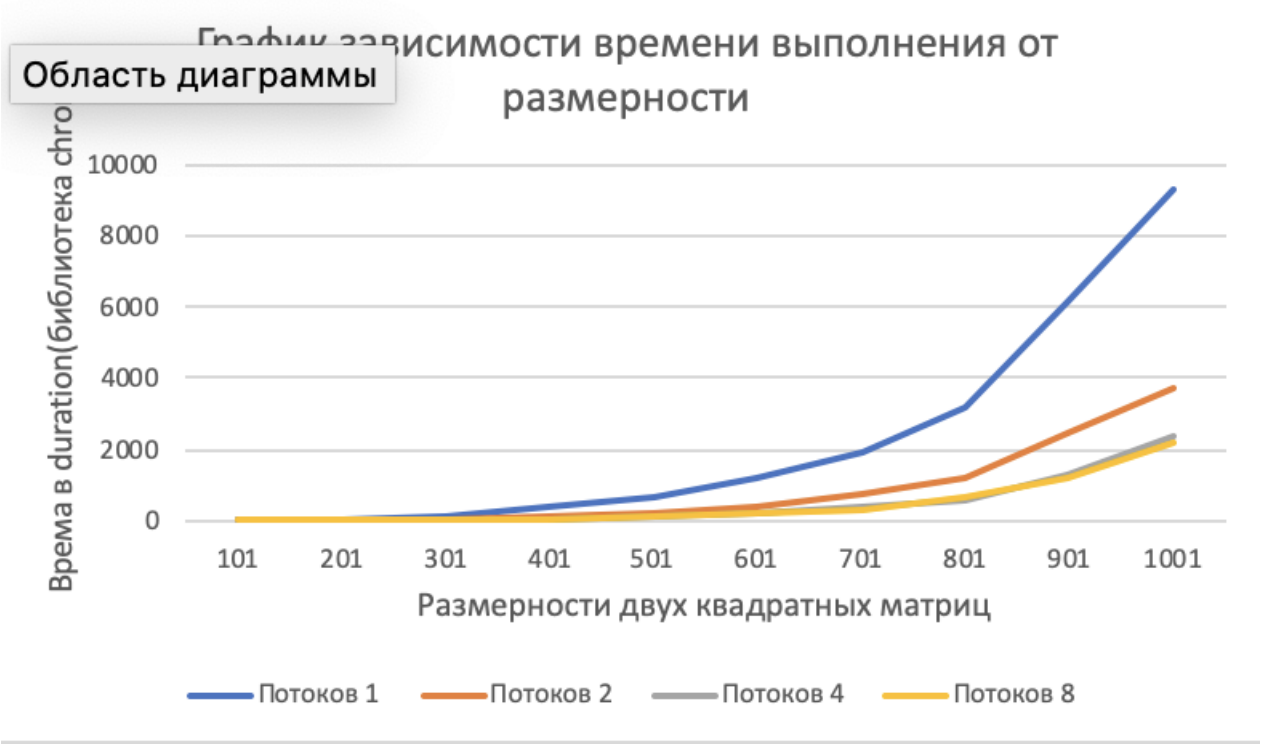

Рис. 4.2: График зависимости времени работы однопоточной и многопоточных версий алгоритма для нечетной размерности матриц

Из данных графиков видно, что присутствие хотя бы двух потоков в несколько раз эффективнее, в сравнению с однопоточной реализацией алгоритма. Разница по времени выполнения в многопоточной реализации алгоритма не зависит от четной или нечетной размерности матриц. Самое опитмальное количество потоков для реализации данного алгоритма - 4, из-за того что на восьми потоках разниц не существенна.

4.2 Вывод

В данном разделе было проведено исследование однопоточной и многопоточных версий алгоритма. Приведены графики зависимостей времени работы алгоритма от размерности матриц.

Среди всех версий алгоритма самыми лучшим оказались версии, где задействовалось не менее четырех потоков. Многопоточная версия алгоритмов показала хороший реазультат относительно однопоточной версии: восьмипоточная реализация алгоритма эффективней однопоточной на 400% при размерности матриц равной 1000.

бла-бла-бла

Заключение

бла-бла-бла

Литература

[1] ..