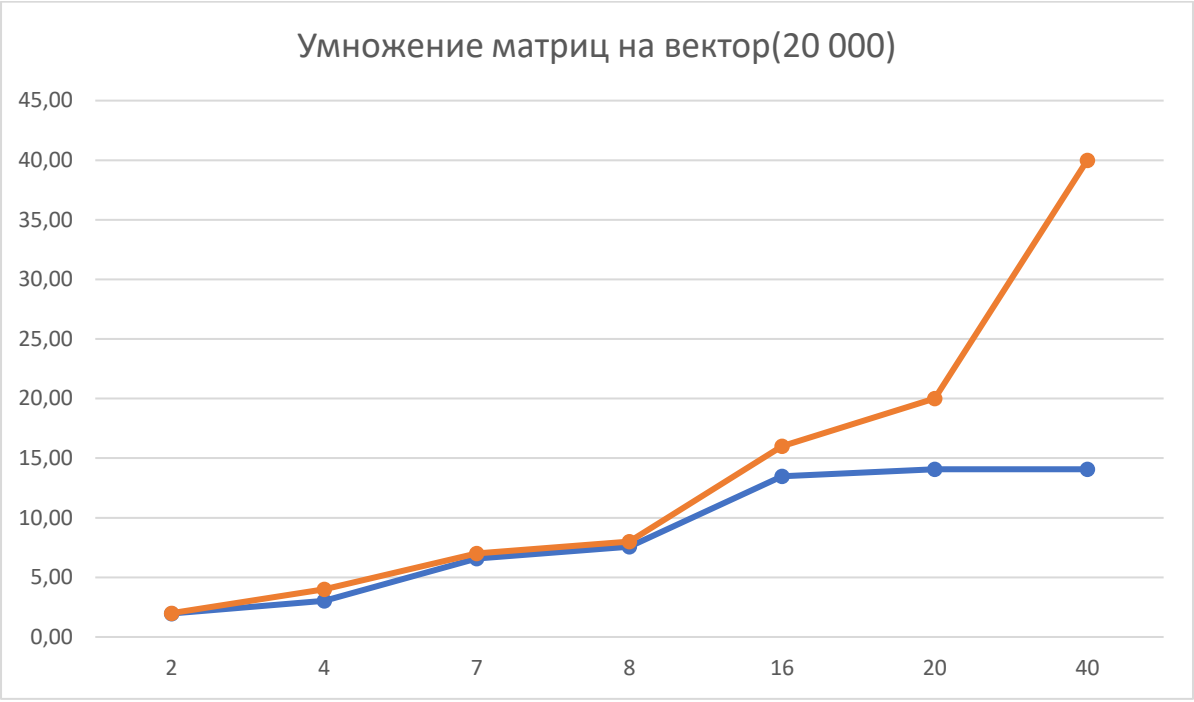
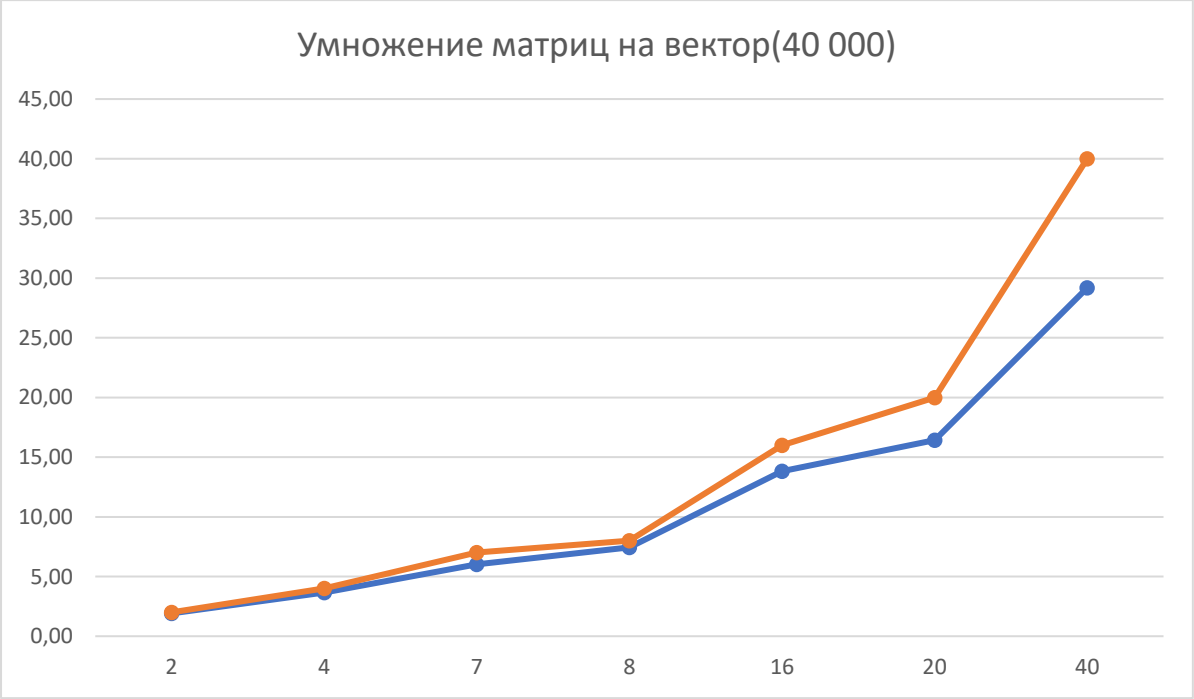
# 1. Умножение матрицы на вектор

| | Количество потоков | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | | 4 | | 7 | | 8 | | 16 | | 20 | | 40 | |
| M=N | T1 | T2 | S2 | T4 | S4 | T7 | S7 | T8 | S8 | T16 | S16 | T20 | S20 | T40 | S40 |
| 20000 | 1,97 | 1,00 | 1,97 | 0,65 | 3,03 | 0,30 | 6,57 | 0,26 | 7,58 | 0,15 | 13,49 | 0,14 | 14,07 | 0,14 | 14,07 |
| 40000 | 7,88 | 4,12 | 1,91 | 2,15 | 3,67 | 1,31 | 6,02 | 1,06 | 7,43 | 0,57 | 13,82 | 0,48 | 16,42 | 0,27 | 29,19 |



На графиках красным указан

При увеличении числа потоков коэффициент ускорения сначала растет почти линейно, но затем его рост замедляется. Это связано с ростом накладных расходов:

1. создание и управление потоками;

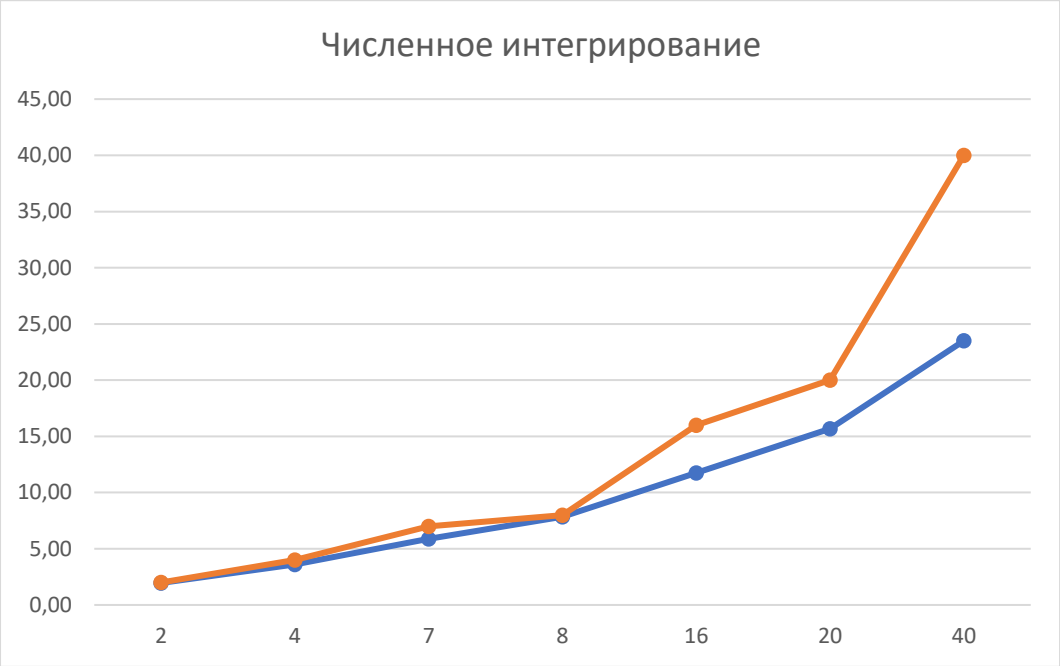
2. синхронизацию доступа к общей памяти;

3. обмен данными между потоками.

Эти расходы начинают доминировать при большом количестве потоков, снижая эффективность. Так же при фиксированном размере задачи и увеличении числа потоков на каждый поток приходится все меньше работы. Если объем вычислений на поток слишком мал, накладные расходы на переключение потоков могут превысить выигрыш от параллелизации.

## 2. Численное интегрирование

| | Количество потоков | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | | 4 | | 7 | | 8 | | 16 | | 20 | | 40 | |
| nsteps | T1 | T2 | S2 | T4 | S4 | T7 | S7 | T8 | S8 | T16 | S16 | T20 | S20 | T40 | S40 |
| 40000000 | 0,47 | 0,24 | 1,96 | 0,13 | 3,62 | 0,08 | 5,88 | 0,06 | 7,83 | 0,04 | 11,75 | 0,03 | 15,67 | 0,02 | 23,50 |




При увеличении числа потоков коэффициент ускорения сначала растет почти линейно, но затем его рост замедляется. Это связано с ростом накладных расходов:

1. создание и управление потоками;

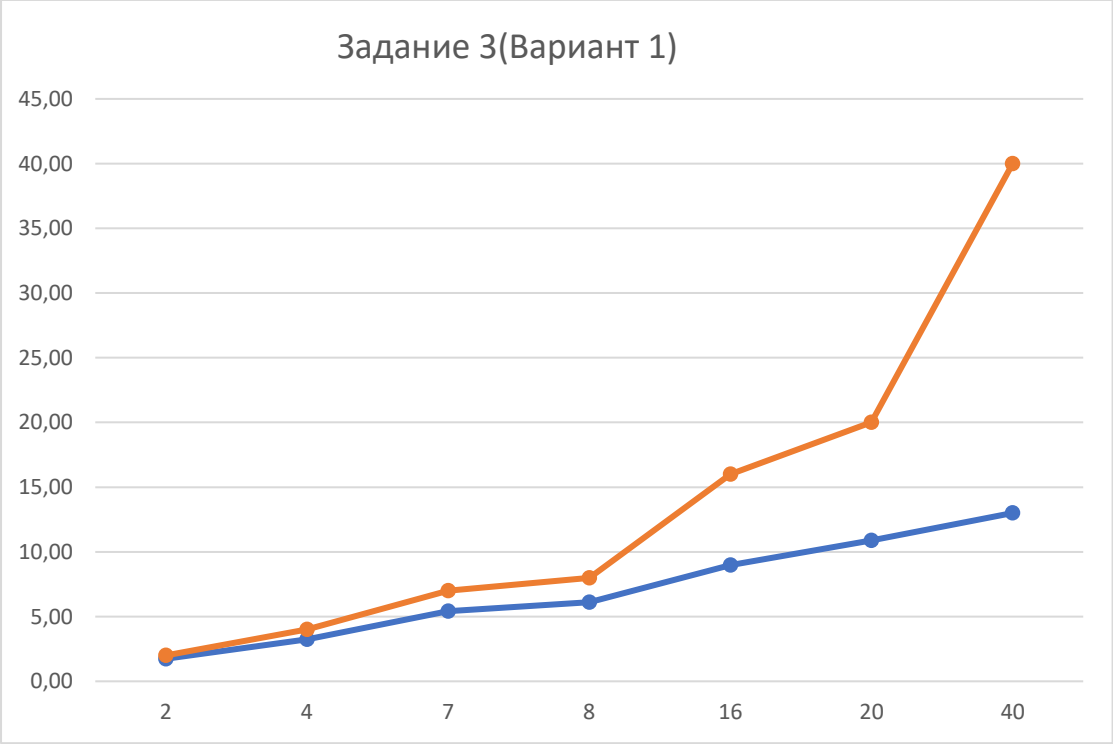
2. синхронизацию доступа к общей памяти;

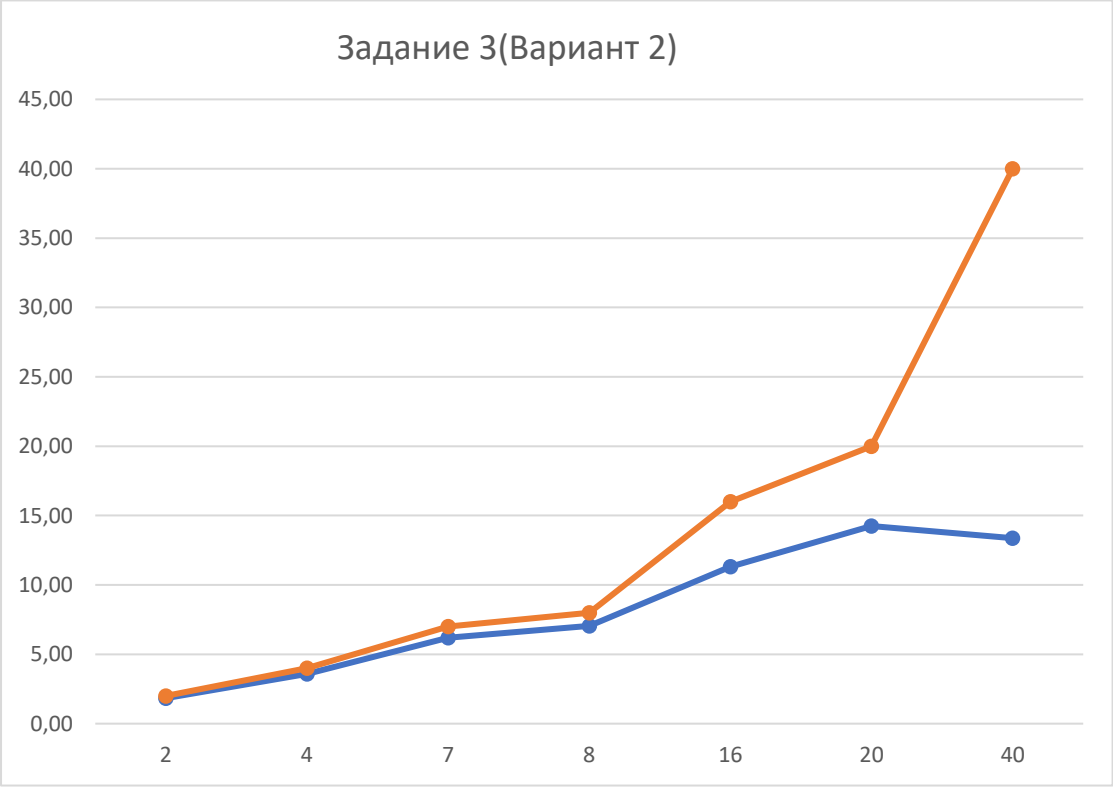
3. обмен данными между потоками.

Эти расходы начинают доминировать при большом количестве потоков, снижая эффективность. Так же при фиксированном размере задачи и увеличении числа потоков на каждый поток приходится все меньше работы. Если объем вычислений на поток слишком мал, накладные расходы на переключение потоков могут превысить выигрыш от параллелизации.

## 3. Итерационный метод

| | Количество потоков | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | | 4 | | 7 | | 8 | | 16 | | 20 | | 40 | |
| | T1 | T2 | S2 | T4 | S4 | T7 | S7 | T8 | S8 | T16 | S16 | T20 | S20 | T40 | S40 |
| 1 вариант | 44,34 | 25,47 | 1,74 | 13,64 | 3,25 | 8,19 | 5,41 | 7,27 | 6,10 | 4,94 | 8,98 | 4,07 | 10,89 | 3,41 | 13,00 |
| 2 вариант | 44,16 | 24,03 | 1,84 | 12,30 | 3,59 | 7,14 | 6,18 | 6,27 | 7,04 | 3,90 | 11,32 | 3,10 | 14,25 | 3,30 | 13,38 |



При увеличении числа потоков коэффициент ускорения сначала растет почти линейно, но затем его рост замедляется. Это связано с ростом накладных расходов:

1. создание и управление потоками;

2. синхронизацию доступа к общей памяти;

3. обмен данными между потоками.

Эти расходы начинают доминировать при большом количестве потоков, снижая эффективность. Так же при фиксированном размере задачи и увеличении числа потоков на каждый поток приходится все меньше работы. Если объем вычислений на поток слишком мал, накладные расходы на переключение потоков могут превысить выигрыш от параллелизации.

Исходя из замеров времени и коэффициентов ускорения, можно сделать вывод, что второй вариант реализации задачи, предполагающий создание одной параллельной секции **#pragma omp parallel**, охватывающей весь итерационный алгоритм, выгоднее с точки зрение производительности, чем создание отдельных параллельных секций **#pragma omp parallel for**.
Скорее всего, это связано с тем, что каждый **#pragma omp parallel for** создает и завершает новый параллельный регион. Это приводит к частому запуску и остановке потоков, что вносит дополнительную нагрузку. В то время как **#pragma omp parallel** создает один параллельный регион, в рамках которого потоки переиспользуются на протяжении всего выполнения блока.

## 4. Schedule

| schedule_type | time |
|---|---|
| static | 13,35 |
| dynamic | 13,41 |
| guided | 13,39 |

Замеры приведены для chunk_size = 1250. Эффективнее всего оказался static.