

ОТЧЁТ ПО ЭКСПЕРИМЕНТУ (режимы `schedule` в OpenMP)

ОТЧЁТ ПО ЭКСПЕРИМЕНТУ (режимы `schedule` в OpenMP)

1. Описание эксперимента

В работе исследуется влияние различных режимов распределения итераций цикла `for` на производительность OpenMP-программы с заведомо **неравномерной** нагрузкой на итерациях. Рассматриваются три варианта планирования: `schedule(static)`, `schedule(dynamic)` и `schedule(guided)`.

Цикл сконструирован так, что большая часть итераций остаётся относительно «лёгкой», а меньшая часть выполняет существенно больше арифметических операций, создавая неравномерность по времени выполнения. Для оценки масштабируемости выбраны два размера задачи: $N = 5 \times 10^5$ и $N = 5 \times 10^6$; для каждого из них измеряются время выполнения и ускорение относительно последовательного варианта:

$$S(p) = \frac{T_{seq}}{T_{par}(p)}.$$

2. Реализация цикла и параллельных версий

Нагрузка на итерацию задаётся числом внутренних повторений, которое резко возрастает после некоторого индекса i_{heavy} :

$$\text{reps}(i) = \begin{cases} r_{light}(i), & i < i_{heavy}, \\ r_{heavy}, & i \geq i_{heavy}. \end{cases}$$

Последовательный вариант цикла:

```
““c for (long long i = 0; i < N; ++i) { int reps = (i < heavy_start ? light_reps(i) : heavy_reps); double x = 0.0; for (int k = 0; k < reps; ++k) { x = x * 1.0000001 + 1.0; } acc += x; }
```

Параллельная версия использует редукцию по сумме и выбор режима планирования через `omp_set_schedule`:

```
omp_set_schedule(policy, chunk_size);
```

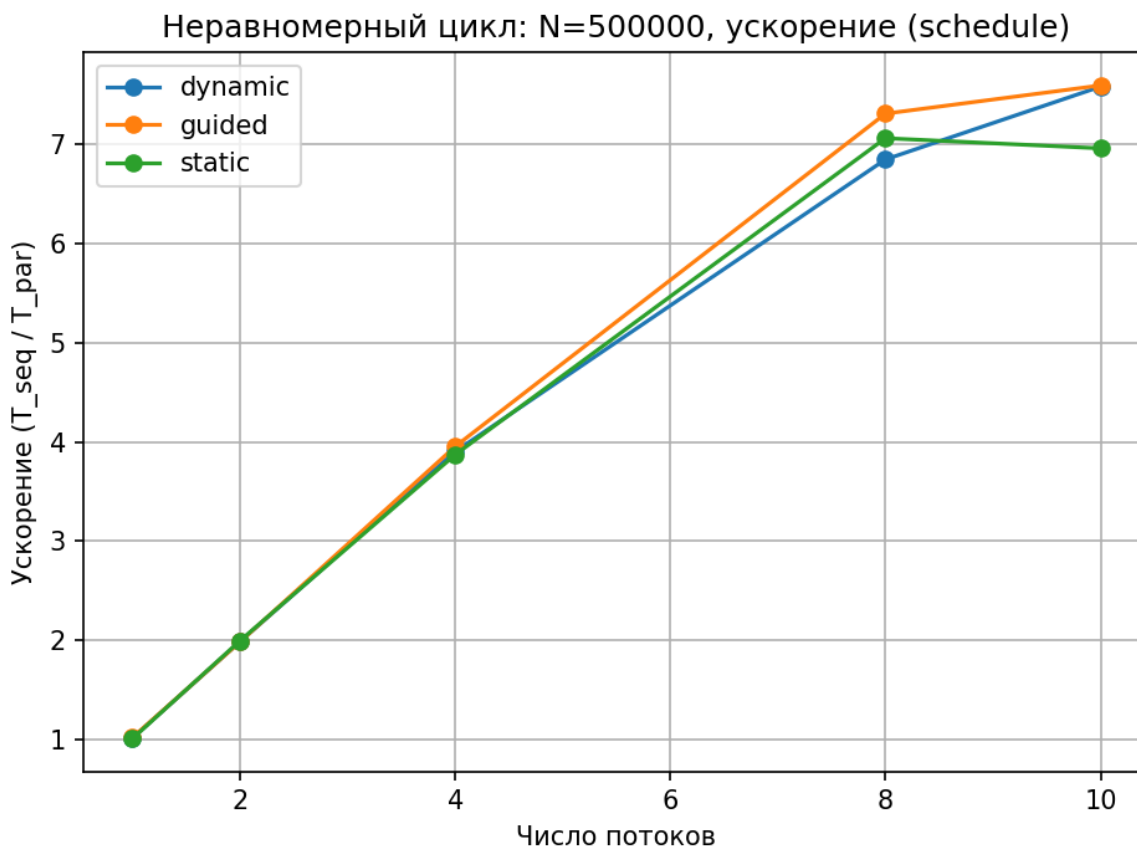
```
#pragma omp parallel for schedule(runtime) reduction(+ : acc) for (long long i = 0; i < N; ++i) { int  
reps = (i < heavy_start ? light_reps(i) : heavy_reps); double x = 0.0; for (int k = 0; k < reps; ++k) {  
x = x * 1.0000001 + 1.0; } acc += x; }
```

При `static` диапазон индексов заранее делится на непрерывные блоки между потоками. При `dynamic` нити получают новые порции работы по мере освобождения. При `guided` размер блоков по ходу выполнения уменьшается, что снижает накладные расходы по сравнению с чисто динамическим вариантом.

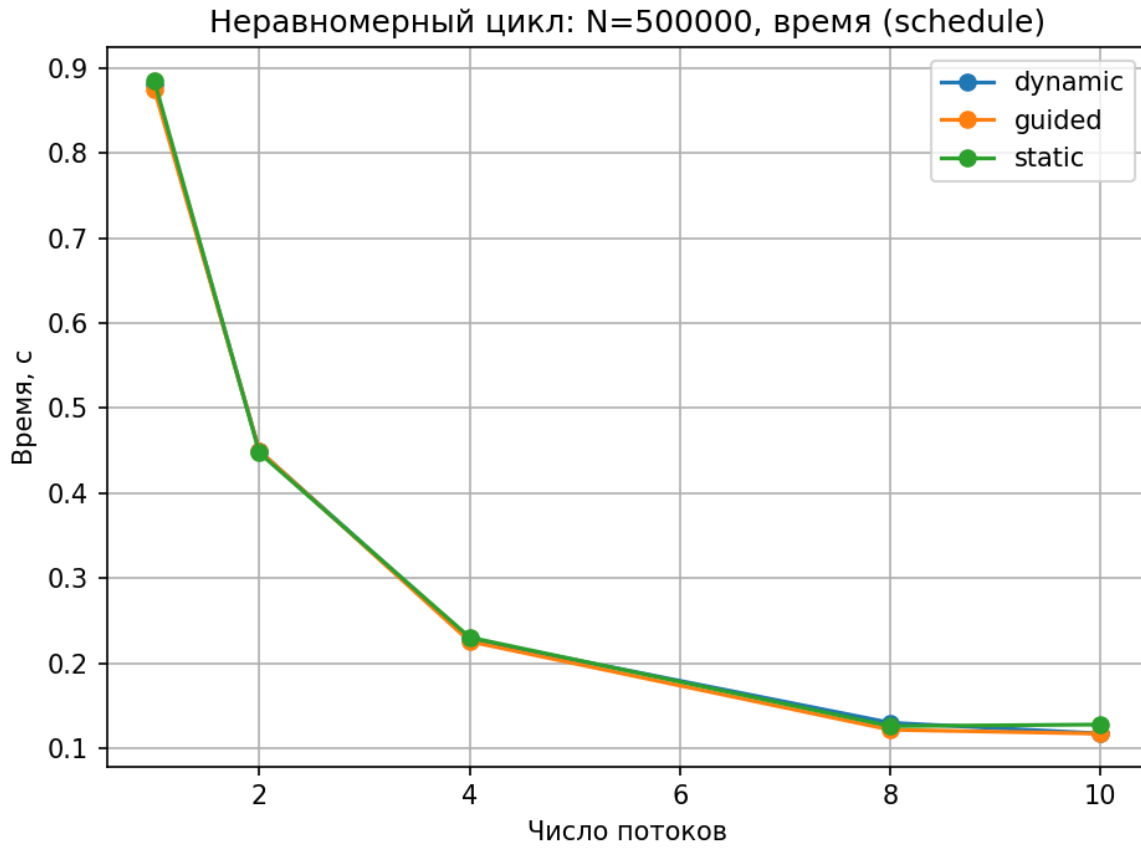
3. Результаты для $N = 5 \cdot 10^5$

3.1. Графики

Ускорение



Время выполнения



3.2. Наблюдения (по графикам)

- На диапазоне 1-10 потоков кривые `static`, `dynamic`, `guided` лежат близко друг к другу. Ускорение растёт почти линейно до 8 потоков и выходит на плато около $S \approx 7-7.5$ при 8-10 потоках.
- Время исполнения уменьшается примерно с ~ 0.9 с на 1 потоке до ~ 0.12 с на 8-10 потоках.
- Различия между режимами планирования невелики; при 8-10 потоках `guided` обычно чуть лучше по времени, а `static` может слегка проигрывать из-за дисбаланса (эффект небольшой).

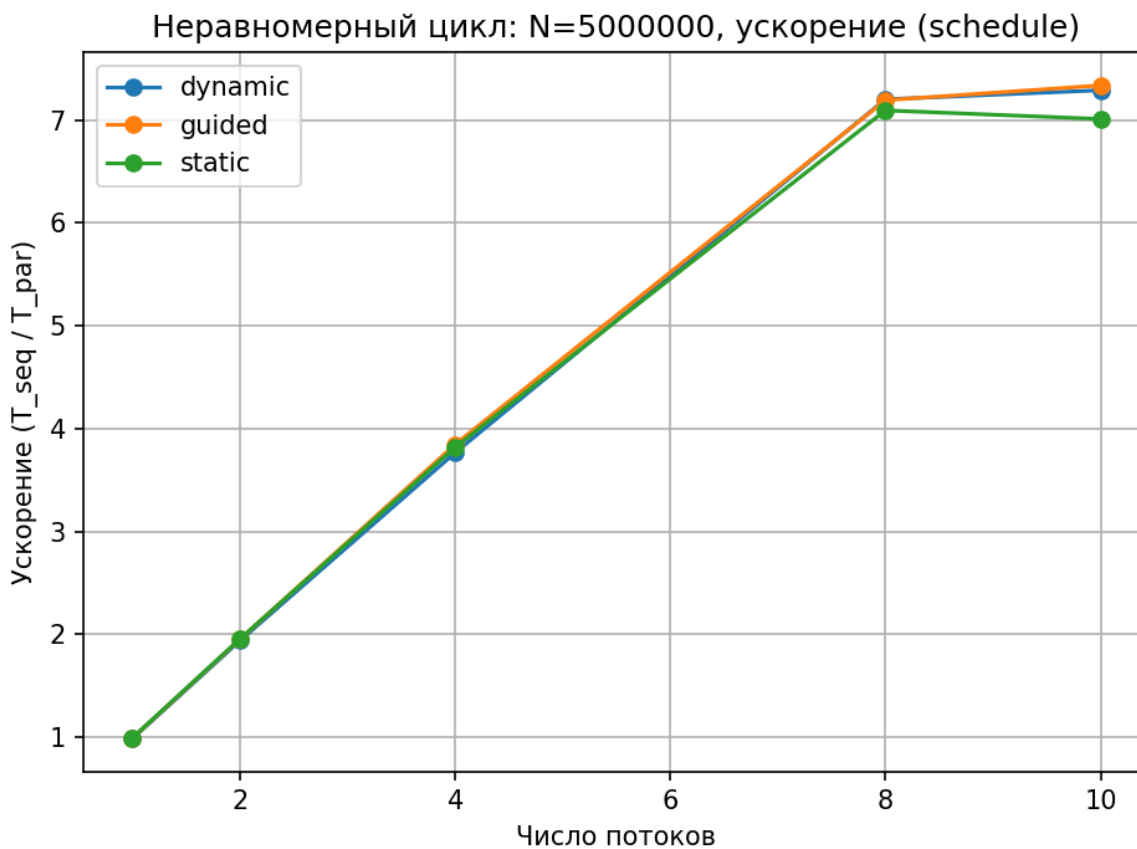
3.3. Объяснение

При $N = 5 \cdot 10^5$ абсолютный объём вычислений относительно невелик, а доля «тяжёлых» итераций не создаёт сильного перекаса нагрузки. Накладные расходы на более сложное планирование (`dynamic/guided`) сопоставимы с потенциальным выигрышем от балансировки, поэтому все три режима демонстрируют почти одинаковую производительность.

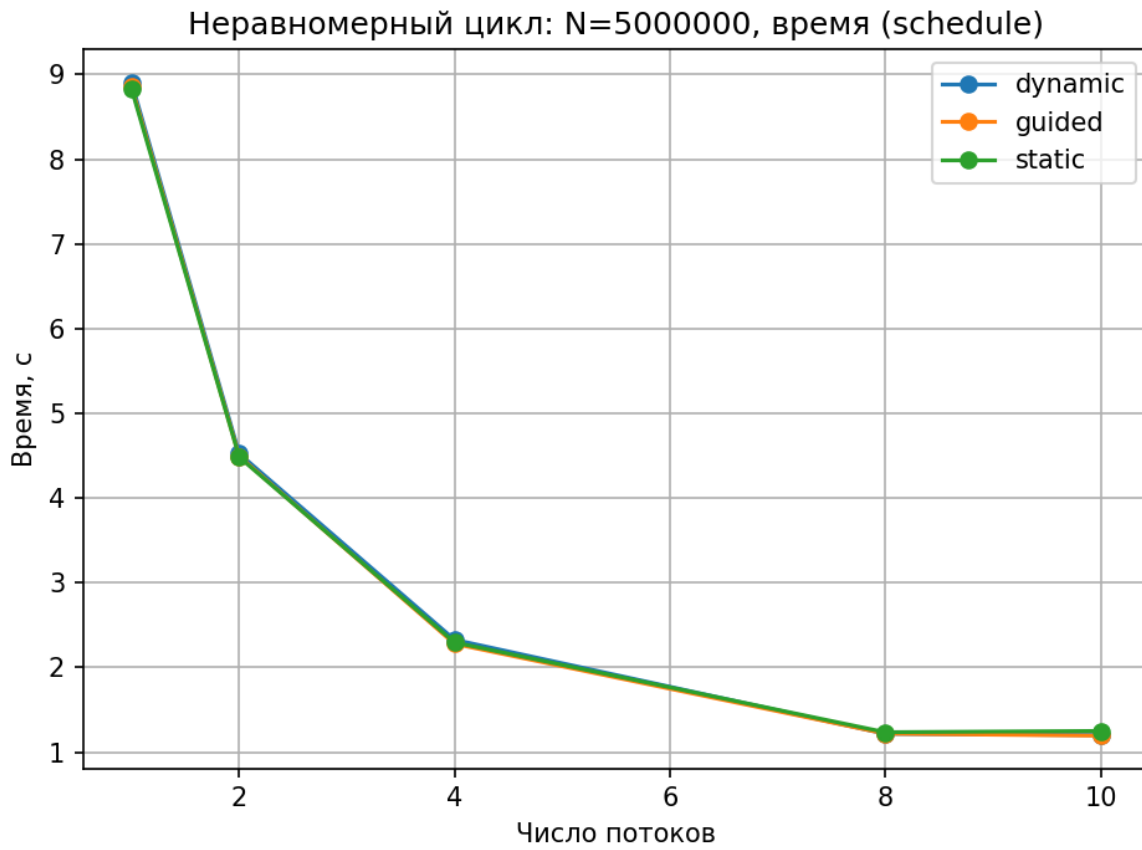
4. Результаты для $N = 5 \cdot 10^6$

4.1. Графики

Ускорение



Время выполнения



4.2. Наблюдения (по графикам)

- Время выполнения на 1 потоке составляет около 8.8-9 с; при 8-10 потоках уменьшается до ~1.2-1.3 с.
- Ускорение достигает $S \approx 7-7.3$; профили `static`, `dynamic`, `guided` очень близки и отличаются лишь на несколько процентов.
- На 8-10 потоках `guided` обычно немного выигрывает по времени, `dynamic` близок к нему, а `static` может слегка уступать из-за неравномерности распределения «тяжёлых» итераций.

4.3. Объяснение

Даже при увеличении размера задачи структура нагрузки остаётся прежней, а «тяжёлые» итерации не приводят к ситуации, когда весь тяжёлый участок достаётся одному потоку при статическом распределении. Поэтому ни один режим `schedule` не даёт выраженного преимущества, а масштабируемость ограничивается в основном накладными расходами OpenMP и аппаратными факторами (например, пропускной способностью памяти/конкуренцией за ресурсы).

##5. Итоговые выводы Для выбранного примера неравномерного цикла все три режима `schedule(static)`, `schedule(dynamic)` и `schedule(guided)` демонстрируют очень близкую произ-

водительность на диапазоне от 1 до 10 потоков. При обоих размерах задачи достигается ускорение порядка ($S \approx 7-7.5$), а после ~ 8 потоков наблюдается насыщение (плато) ускорения. В данной конфигурации накладные расходы динамического/направляемого планирования не дают заметного выигрыша, поэтому static остаётся конкурентным, а guided выглядит лишь слегка предпочтительнее на больших числах потоков.