Чигарев Дмитрий 381807-1

Лабораторная работа 1.9

«Выигрыш в производительности от использования потоков»

Потоки в языке C# исполняются параллельно, что говорит о том, что в многопроцессорных системах можно получить прирост в производительности при грамотном их использовании.

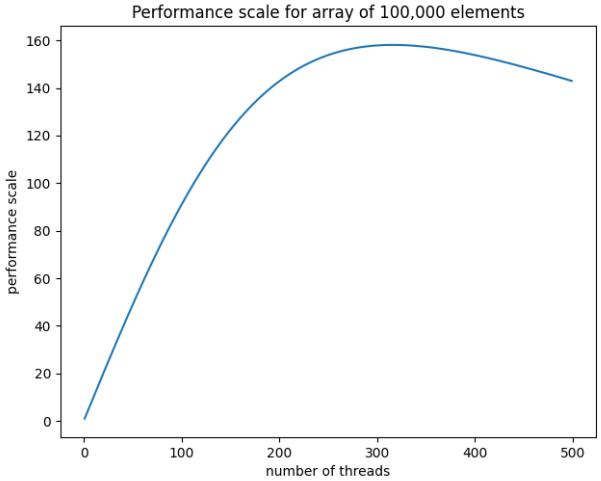
В качестве примера напишем параллельную сортировку пузырьком: массив, который нужно будет отсортировать будет разбиваться на N частей, каждая часть отправляется в свой поток на сортировку пузырьком, после чего из результатов всех частей формируется конечный массив путем их слияния с учетом порядка. Асимптотическая сложность данного подхода будет составлять:

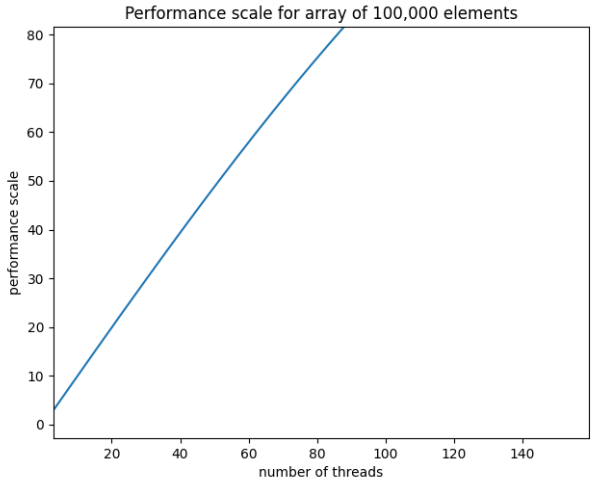
$$(N/K)^2 + NK, \qquad \text{где } K - \text{число потоков}$$

При этом:

$$\forall K > 1 \rightarrow \exists N\colon\ N^2 > (N/K)^2 + NK$$

Т.е. при числе потоков равном двум мы в теории уже получим прирост в производительности, а коэффициент роста скорости будет равен: $NK/(N + K^2)$

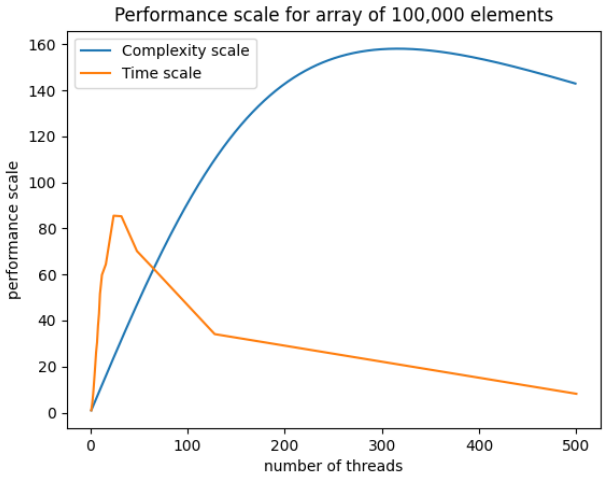


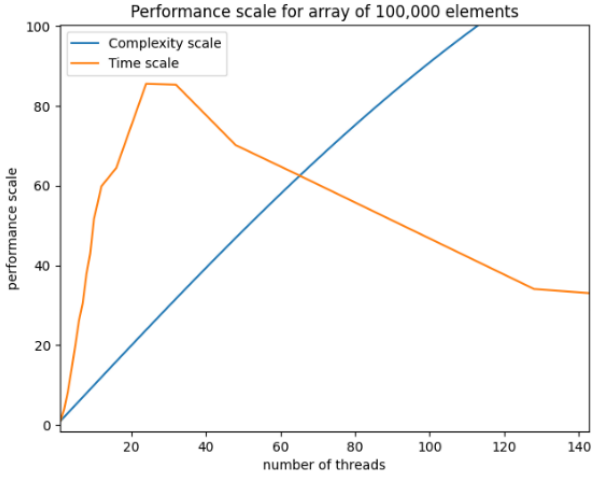


Но это всё лишь асимптотические оценки, посмотрим, что будет на практике. Реализуем описанный выше алгоритм и сделаем серию замеров производительности по следующей методике:

1. Будем измерять время исполнения функции сортировки.
2. Замеры для каждого числа потоков будут сделаны 5 раз подряд, результатом будет считаться медиана полученных измерений
3. Массив, который нужно отсортировать, в каждом замере один и тот же, его длинна 100,000 элементов.
4. Замеры будут проведены для числа потоков $1..128$
5. Процессор: Intel Core i7 8750H 2.2HZ (12 cores)

Замеры сделаны, теперь отобразим результаты на графике. Для начала сравним увеличение скорости относительно числа операций и относительно фактического времени исполнения алгоритма:



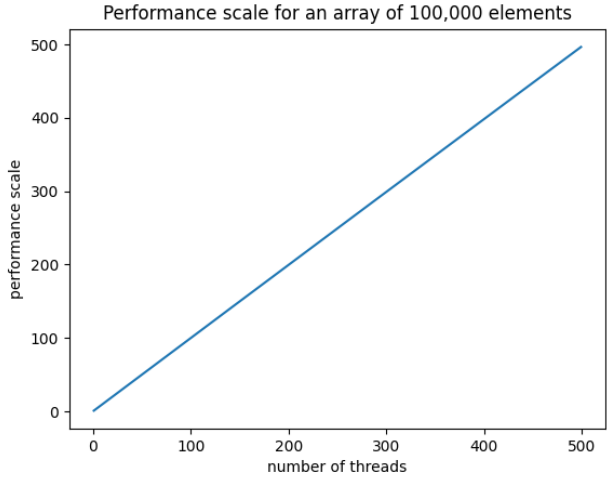


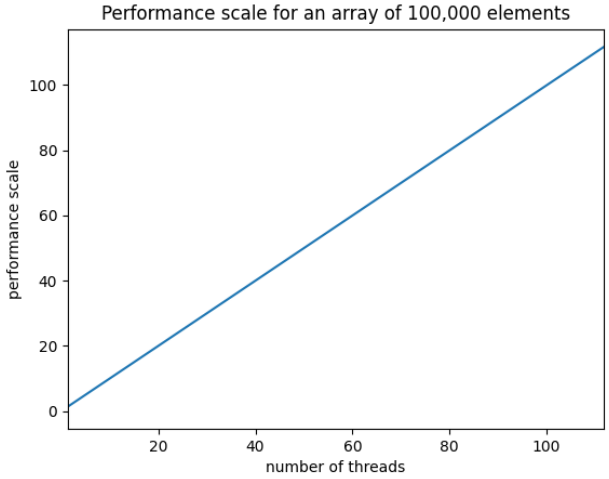
Как видно, уменьшение времени исполнения с ростом потоков происходит гораздо стремительнее, если сравнивать с уменьшением числа операций. Однако, как только кол-во потоков начинает превышать фактическое кол-во ядер в системе примерно в два раза, то рост сменяется спадом.

Попробуем уменьшить сложность фазы слияния, разбив массив не по индексам, а по значениям (более подробно об алгоритме см. в [задании к работе]). Тогда асимптотическая сложность фазы слияния будет составлять N вместо KN, и соответственно итоговая сложность:

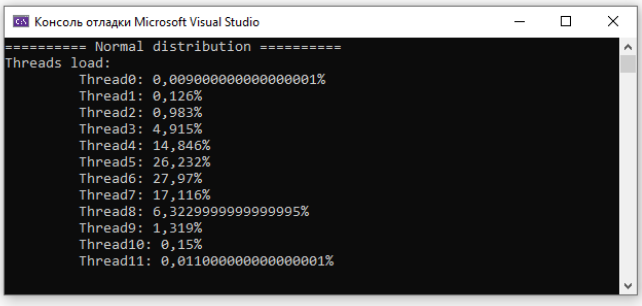
$$(N/K)^2 + N, \qquad \text{где } K - \text{число потоков}$$

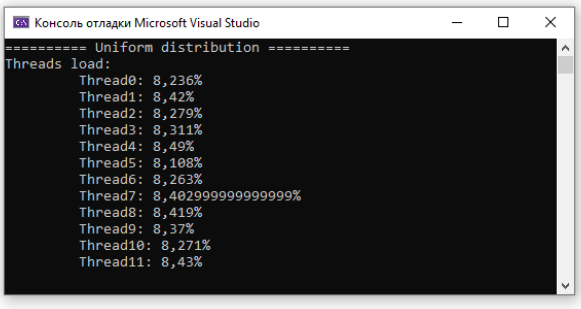
Коэффициент роста скорости будет равен: $NK/(N+K)$





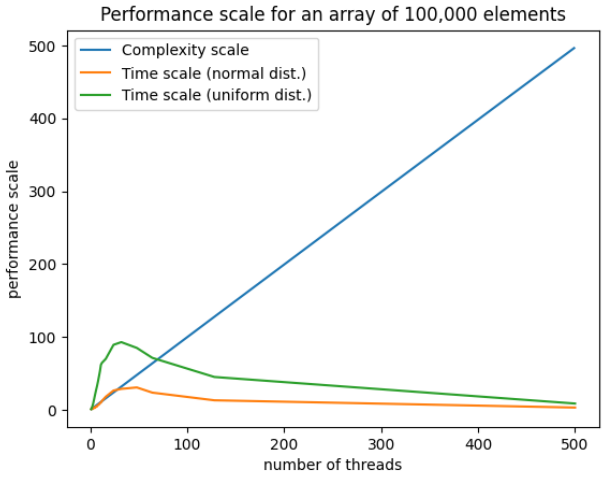
Однако следует отметить, что таких показателей мы сможем достигнуть лишь при равномерном распределение данных по потокам, что будет достигнуто при равномерном распределении значений в исходном массиве. При нормальном распределении, большинство потоков оказываются незагруженными:

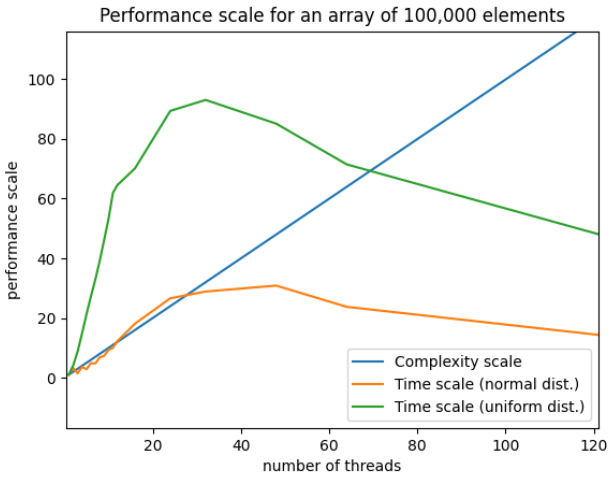




Поэтому будем в этот раз будем исследовать два случая: при нормальном распределении элементов в массиве и при равномерном.

Замеры показали следующие результаты:





Приложения

1. Исходный код программы: https://github.com/proxodilka/csharp-labs/tree/master/blab9