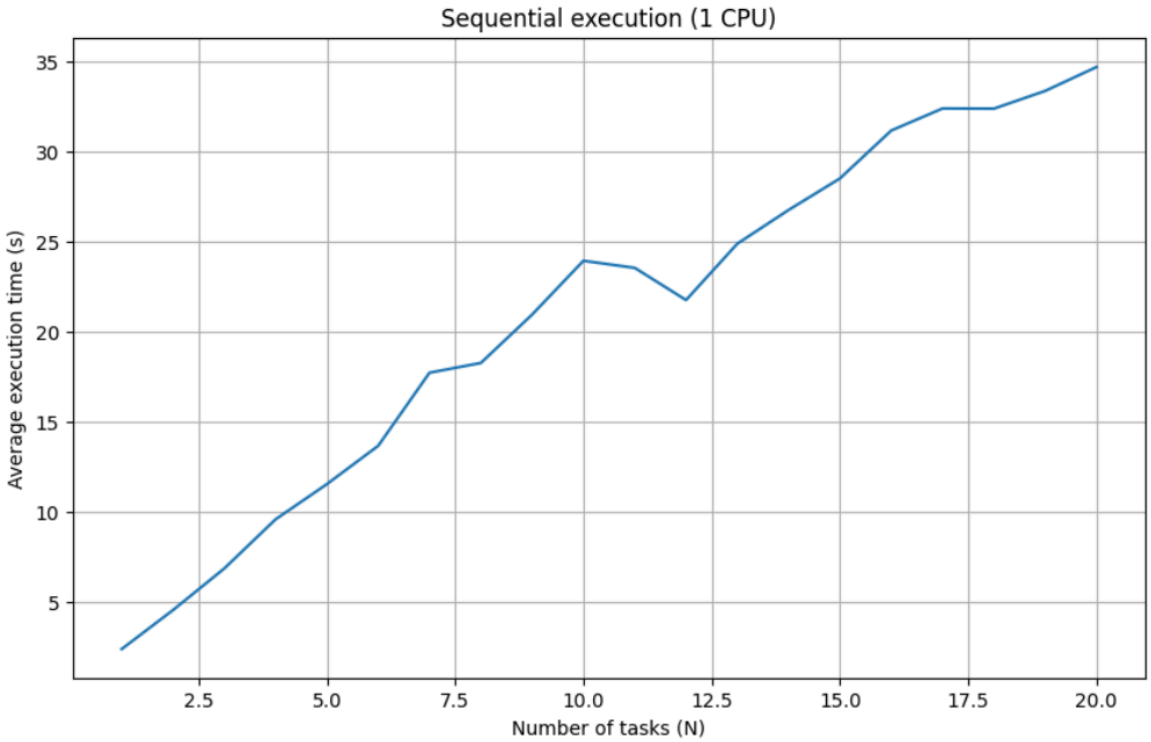
1. Эксперименты с последовательным и параллельным выполнением вычислительно сложных задач

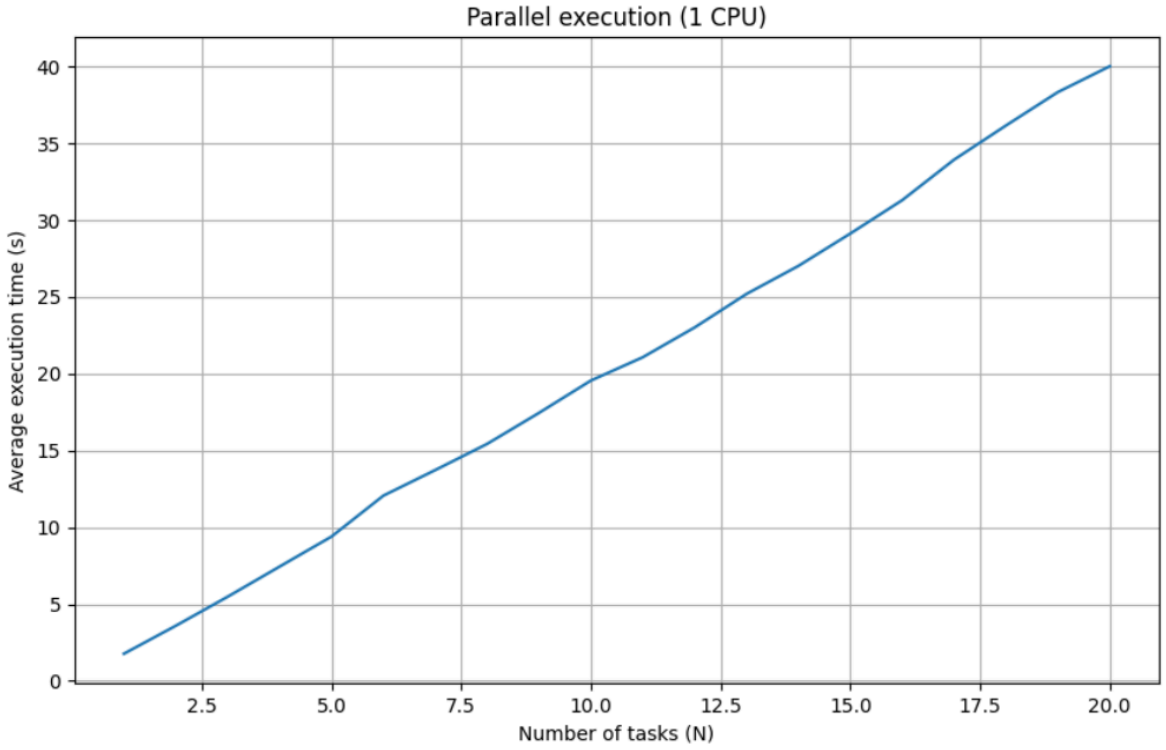
Для выполнения этой группы экспериментов использовался алгоритм для вычисления суммы ряда для числа $\frac{\pi^2}{6}$, замеряющий время выполнения.
Для вычисления приближенного значения использовалось 1500000000 итераций, благодаря чему в среднем алгоритм работает за 2,2-2,3 секунды.

Скрипты sequential.sh и parallel.sh с параметром N запускают N выполнения алгоритма (последовательных или параллельных соответственно).
Скрипт exp.sh принимает в качестве параметров:

- mode - режим запуска
- start_N - начальное значение N
- end_N - конечное значение N
- repeats - количество повторений
- CPU - количество процессоров(используется для именования файлов с результатами)

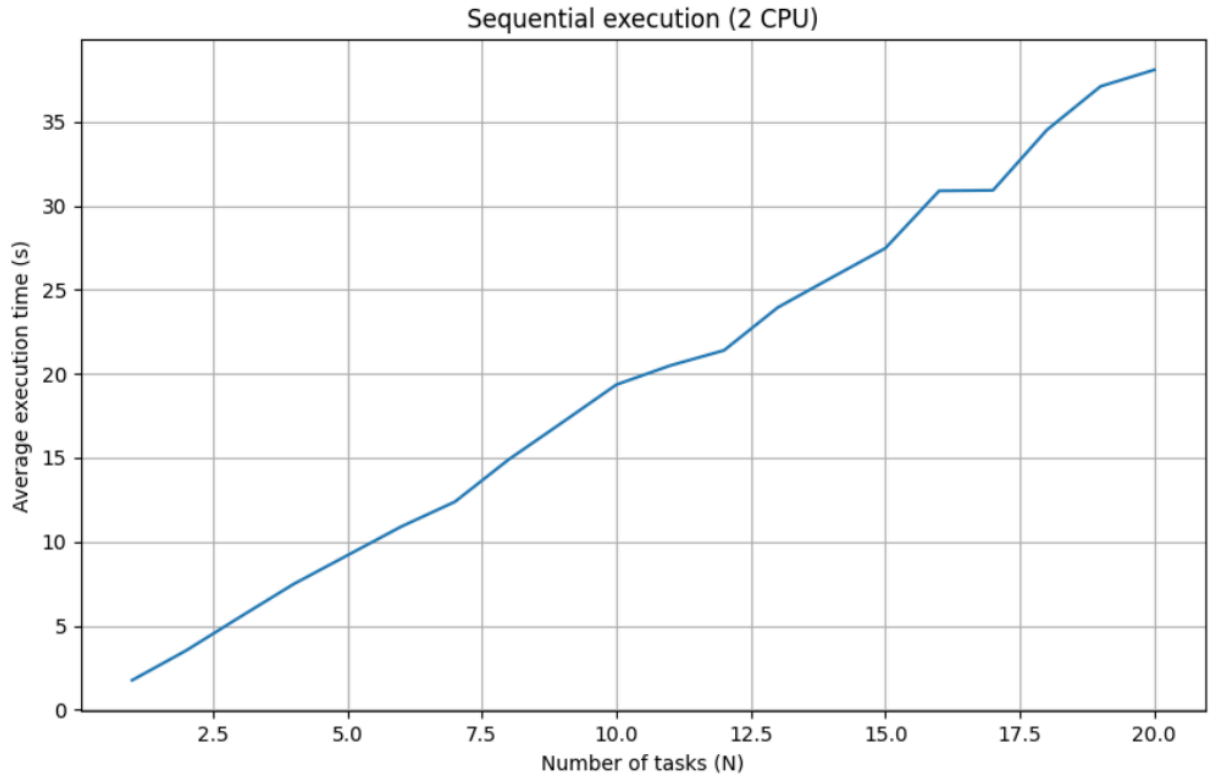
exp.sh последовательно или параллельно(в зависимости от "mode") запускает от "start*N" до "end_N" (в экспериментах от 1 до 20) "repeats" раз (в экспериментах 10) алгоритм, результаты записываются в файл {mode}* results _ {CPU}.csv.
Графики построены на основе этих файлов с результатами:
1 процессор:

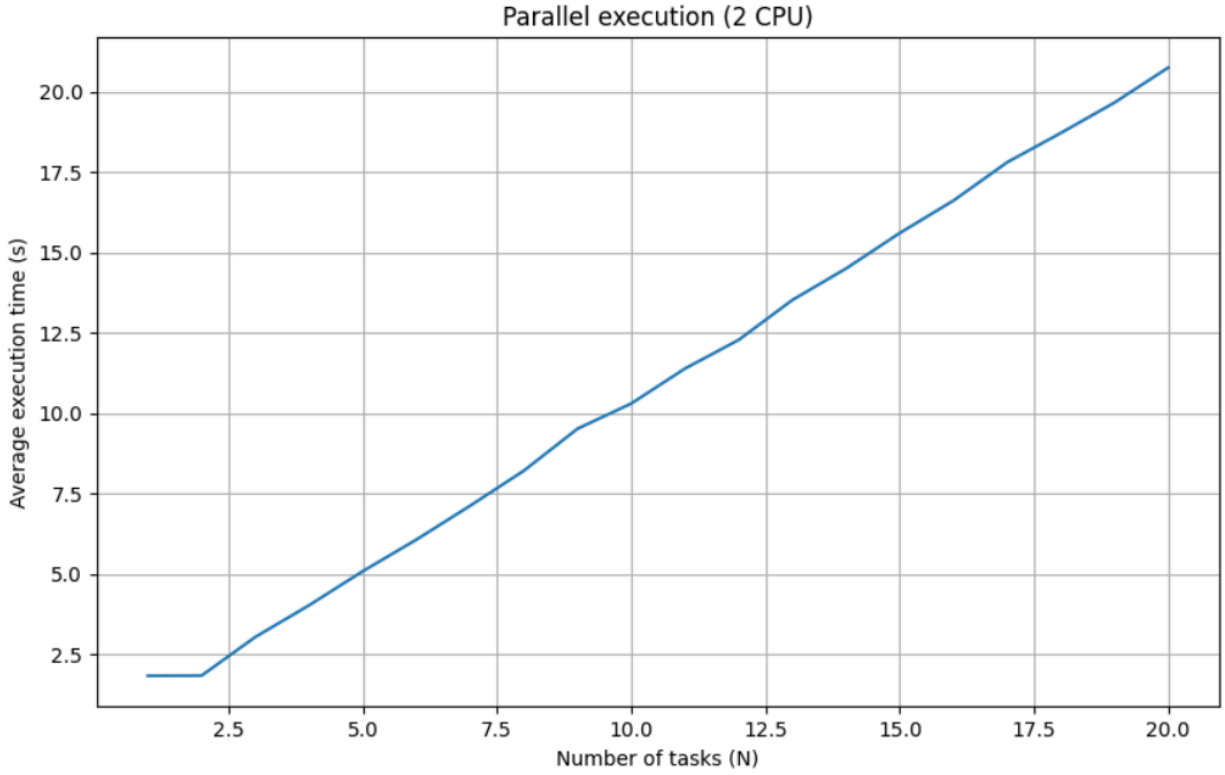




Как и при последовательном выполнении, при параллельном выполнении наблюдается практически линейная зависимость, время увеличивается пропорционально росту задач. На 1 процессоре при параллельном запуске задачи не выполняются параллельно, они делят время CPU и из-за накладных расходов в виде переключения контекста и синхронизации среднее время выполнения даже увеличилось, по сравнению с последовательным.

2 процессора:





На 2-х процессорах последовательное выполнение всё ещё занимает примерно столько же времени(~36 секунд), сколько при одном процессоре, так как доп. ядра не используются при последовательное выполнении.
Параллельное выполнение за счёт использования дополнительного процессора даёт

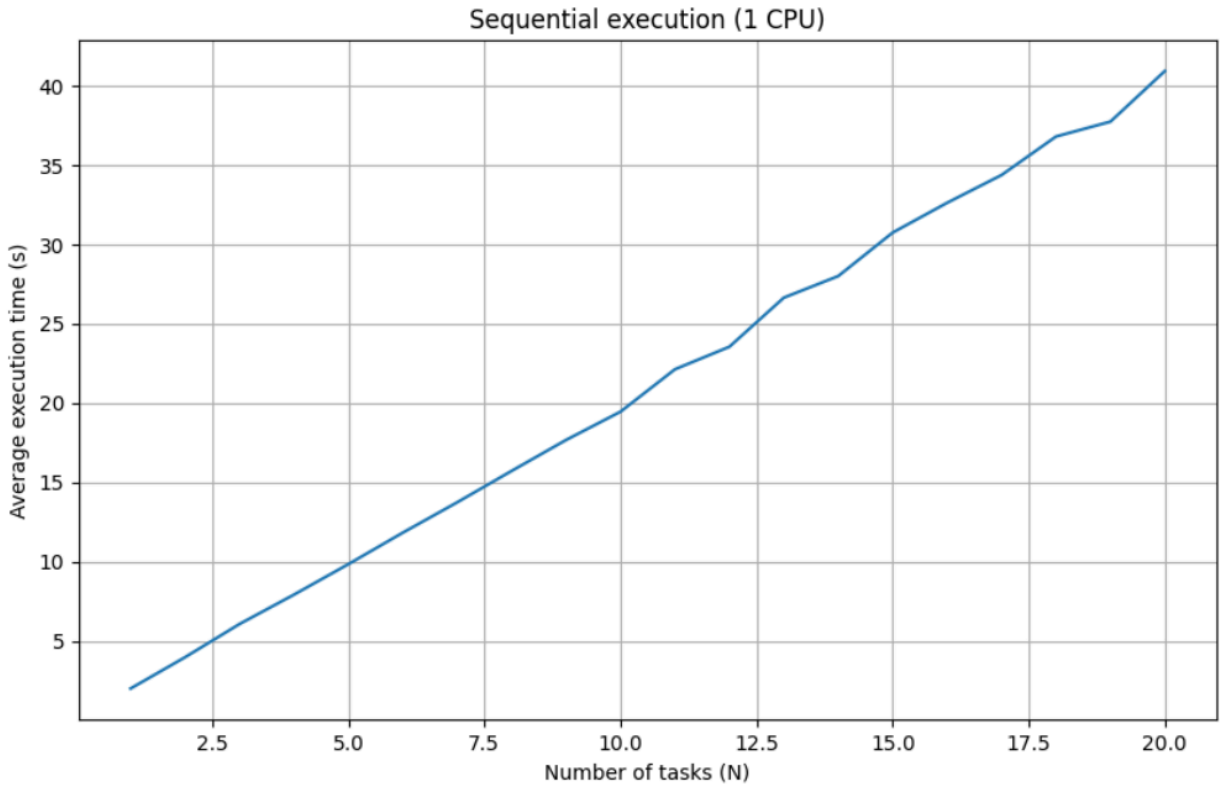
выгоду почти в два раза. Но всё ещё включает в себя дополнительные расходы на переключение контекста и синхронизацию.

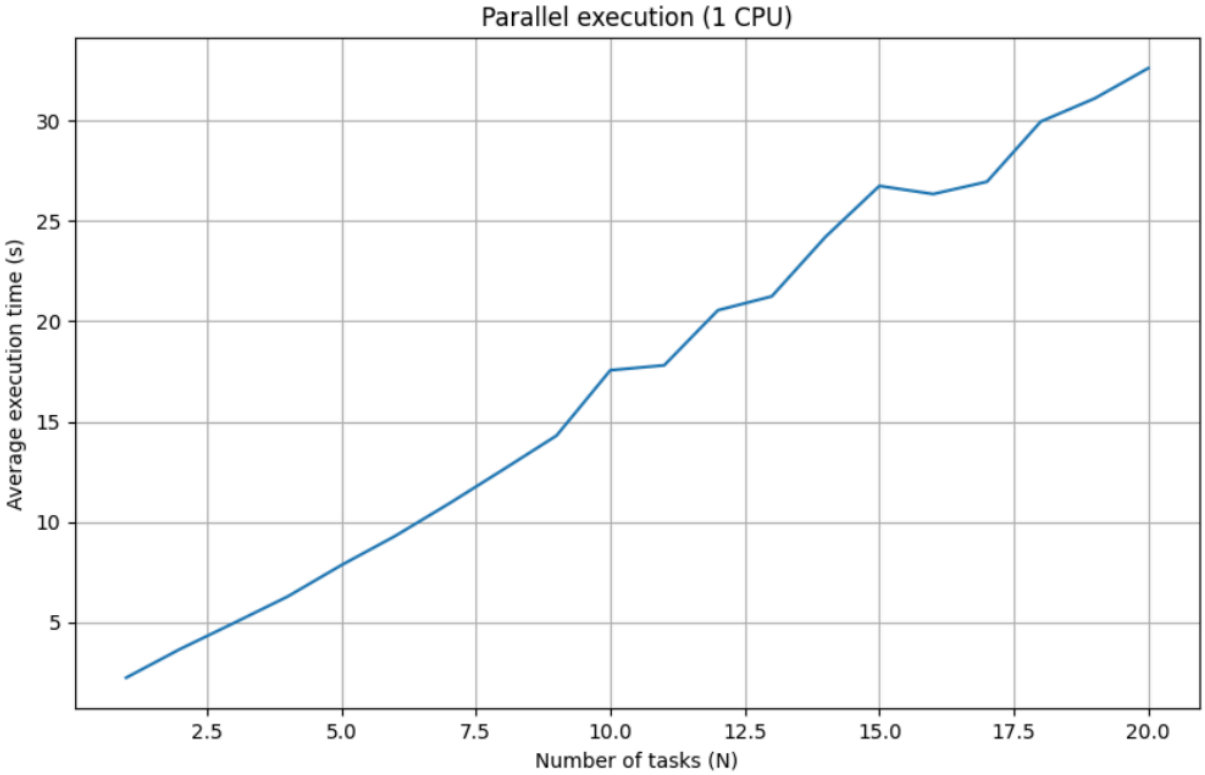
2. Эксперимент с параллельным и последовательным выполнением задач с большими объемами считываемых и сохраняемых данных

Для выполнения этой группы экспериментов использовался алгоритм для обработки данных из файлов размером ~4 MB, благодаря чему в среднем алгоритм работает за 2,2-2,4 секунды.

Скрипты для запусков аналогичные предыдущим экспериментам.

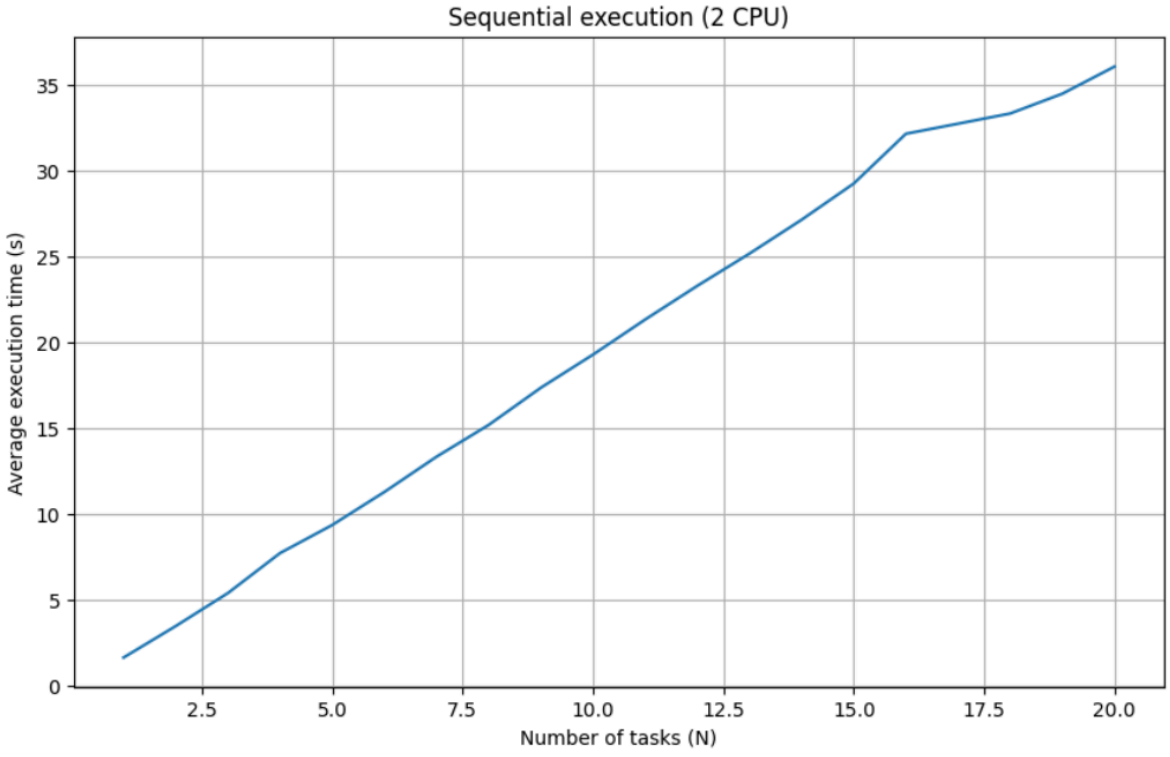
Результаты в виде графиков:
1 процессор:

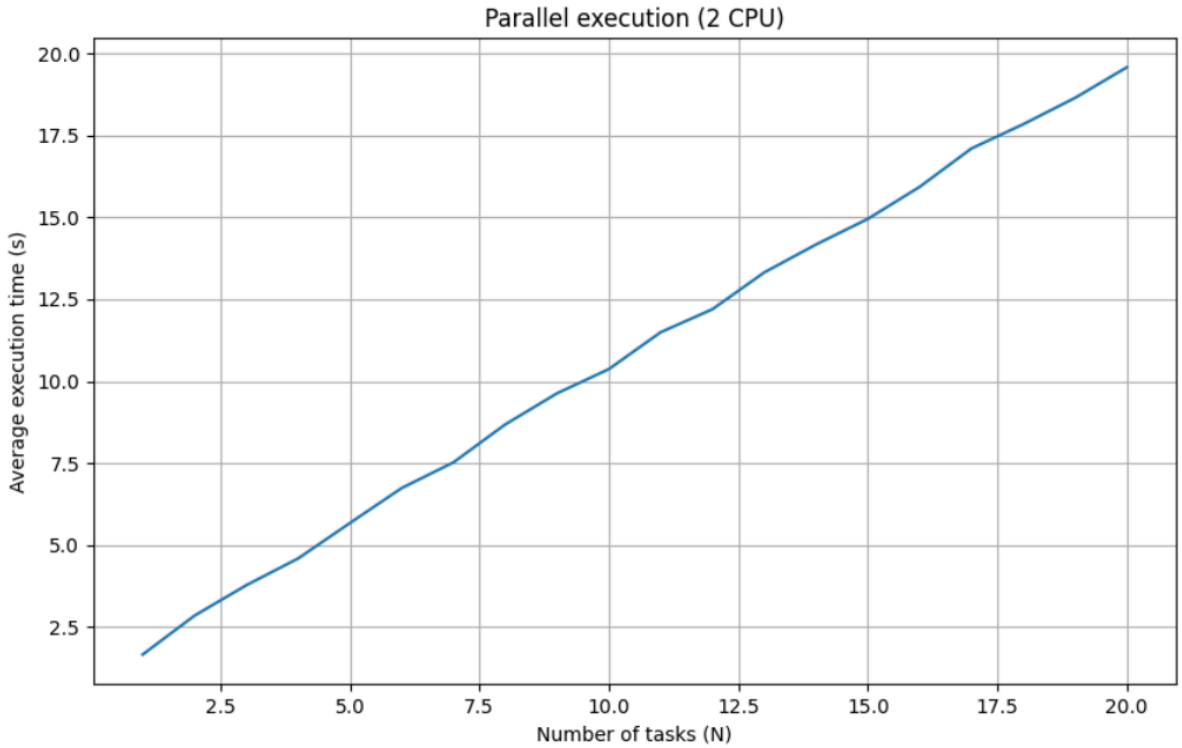




При последовательном запуске время выполнения растет линейно с ростом N. Параллельный режим запуска возможно быстрее, за счет кэширования.

2 процессора:





При параллельном выполнении с двумя процессорами время сокращается почти в два раза. Но так же, как и в предыдущем эксперименте есть дополнительные накладные затраты, а также I/O-ограничения.