Tests

In this lesson, we'll be running tests to see if the parallel version of our app is better than the serial version.

WE'LL COVER THE FOLLOWING

We can run the two versions on a set of files and compare if the changes brought any improvements in the performance.

> Our applications access files, so it's harder to make benchmark correct as we can quickly end up in the file system cache. Before major runs of applications, a tool called Use SysInternal's RAMMap app is executed to remove files from the cache. There are also Hard Drive hardware caches which are harder to release without a system reboot.

The application runs on a 6 core/12 thread machine - i7 8700, with a fast SSD drive, Windows 10.

Mid Size Files 1k Lines 10 Files

Let's start with 10 files, 1k lines each, files are not in the OS cache:

Step	Serial (ms)	Parallel (ms)
All steps	74.05	68.391
`CalcTotalOrder`	0.02	0.22

Parsing Strings	7.85	2.82

The situation when files are in the system cache:

Step	Serial (ms)	Parallel (ms)
All steps	8.59	4.01
`CalcTotalOrder`	0.02	0.23
Parsing Strings	7.74	2.73

The first numbers - 74ms and 68ms - come from reading uncached files, the next two runs were executed without clearing the system cache so you can observe how much speed-up you get by system caches.

The parallel version still reads files sequentially, so we only get a few milliseconds better. Parsing strings (line split and conversion to Records) is now almost 3x faster. The sum calculations are not better as a single-threaded version seem to handle sums more efficiently.

Large Set 10k Lines in 10 Files

How about larger input?

Uncached files:

Step	Serial (ms)	Parallel (ms)
All steps	239.96	178.32
`CalcTotalOrder`	0.2	0.74
Parsing Strings	70.46	15.39

Cached:

Step	Serial (ms)	Parallel (ms)
All steps	72.43	18.51
`CalcTotalOrder`	0.33	0.67
Parsing Strings	70.46	15.56

The more data we process, the better results we get. The cost of loading uncached files “hides” slowly behind the time to process the records. In the case of 10k lines, we can also see that parsing strings step is 3.5 faster; however, the calculations are still slower.

Largest Set 100k Lines in 10 Files

Let’s do one more test with the largest files:

Uncached files:

Step	Serial (ms)	Parallel (ms)
All steps	757.07	206.85
`CalcTotalOrder`	3.03	2,47
Parsing Strings	699.54	143.31

Cached:

Step	Serial (ms)	Parallel (ms)
All steps	729.94	162.49

`CalcTotalOrder`	3.05	2.16
Parsing Strings	707.34	141.28

In a case of large files (each file is ~2MB), we can see a clear win for the parallel version.

In the next lesson, we will wrap up our discussion on parallelization.