Lab 5 Report

Daniel Dohnalek

CPE 315

Data Tables

Original Code	**-O0**	**-O1**	**-O2**	**-O3**
Runtime	202.585 secs	160.178 secs	158.584 secs	170.227 secs
Cache Misses	1,084,546,497	1,081,176,197	1,080,755,968	1,080,409,921
Cache References	22,648,383,722	3,263,648,280	3,261,181,143	3,260,194,574

With best compiler optimization level	**Original**	**Column-major**	**Tiled 16x16**	**Tiled 32x32**
Runtime	158.584 secs	12.841 secs	11.942 secs	14.805 secs
Cache Misses	1,080,755,968	4,094,649	142,419,250	121,946,589
Cache References	3,261,181,143	3,265,725,385	3,598,789,078	3,363,997,373

Explanation

Cache references and cache misses both impact performance. The higher the number of cache references, the longer the program takes to execute. In the -O1 version of the program, there were ~7 times fewer cache references than the -O0 program and a speedup of about 26%. When cache misses decreased by a factor of ~264 due to the column-major optimization, execution time decreased by an impressive 12.3 times. The tiling operation was faster despite the higher number of cache misses because all the matrix blocks that were being operated upon at any given time were able to fit into the cache.

The manual optimization was far more effective than even the compilers most aggressive -O3 option. While the compiler is capable of performing very complex optimizations, in this case it doesn't have enough context about what our code is actually doing to optimize our code ideally.