Cole Brausen(918-418-987)
Tuesday November 19th 2024
CSC-656-01

Coding Project 3

Tables:

	Runtime Data						
Problem Size	blas	basic	vectorized	omp-1	omp-4	omp-16	omp-64
1024	0.00048	0.00352	0.00045	0.03933	0.00155	0.00261	0.00915
2048	0.00051	0.01420	0.00092	0.00463	0.00214	0.00171	0.00192
4096	0.00396	0.05710	0.00414	0.00936	0.00537	0.00330	0.00378
8192	0.01730	0.22891	0.01718	0.02684	0.01782	0.01300	0.01355
16384	0.07145	0.91772	0.06938	0.06020	0.06494	0.05586	0.05776
	MFLOP/s						
Problem Size	blas	basic	vectorized	omp-1	omp-4	omp-16	omp-64
1024	4371.20	596.07	4662.61	53.35	1353.66	803.90	229.31
2048	16452.27	590.89	9120.28	1812.24	3920.87	4906.82	4370.13
4096	8474.38	587.72	8105.92	3585.31	6249.26	10169.25	8877.92
8192	7758.72	586.37	7812.92	5000.97	7532.32	10325.07	9905.97
16384	7514.17	585.02	7738.36	8918.39	8267.44	9611.30	9295.14
	Bandwidth percentage						
Problem Size	blas	basic	vectorized	omp-1	omp-4	omp-16	omp-64
1024	17.08333%	2.32955%	18.22222%	0.20849%	5.29032%	3.14176%	0.89617%
2048	64.28235%	2.30873%	35.63478%	7.08078%	15.31963%	19.17193%	17.07500%
4096	33.10707%	2.29604%	31.66763%	14.00684%	24.41415%	39.72848%	34.68360%
8192	30.30936%	2.29065%	30.52107%	19.53621%	29.42492%	40.33477%	38.69756%
16384	29.35311%	2.28532%	30.22888%	34.83854%	32.29566%	37.54529%	36.31025%

Analysis Questions:

1. At problem size N = 16384, my vectorized implementation's MFLOP/s is approximately 7738.36 compared to basic implementation's 585.02 MFLOP/s which if we take the ratio of makes the vectorized implementation 13.32 times faster than basic. If we look at its bandwidth percentage, then we'll see that vectorized uses 30.22888% of the memory system compared to basic's 2.29% which results in a more frequent memory access by 13.30 times compared to basic which hardly utilizes memory because of its implementation and not using a double in the code.

2. For N = 16384 for the OpenMP implementation that is tested in OpenMP-4 is 8267.44 MFLOP/s compared to basic's 585.02 MFLOP/s which is a 14.13 times increase in MFLOPS/s for OpenMP-4. Memory system utilization indicates a 2.28532% of memory

bandwidth percentage compared to OpenMP-4 32.29566% which indicates a far higher usage in memory in OpenMp-4 compared to basic implementation.

3. For runtime when we go from OpenMP-1 to OpenMP-4, the calculation is:
   0.06020seconds/0.06494 seconds = 0.93 seconds which is a slight slowdown.

   For runtime when we go from OpenMP-1 to OpenMP-16, the calculation is:
   0.06020 seconds/0.05586 seconds = 1.08 seconds.

   For runtime when we go from OpenMP-1 to OpenMP-64, the calculation is:
   0.06020 seconds/0.05776 seconds = 1.04 seconds.

   So in conclusion OpenMP-1 has a slight slowdown when it transitions to OpenMP-4 but it speeds up when it goes to 16. Going from openmp-1 to openmp-64 indicated a modest speedup but nothing massive.