Highlife Assignment 2

Running Instructions

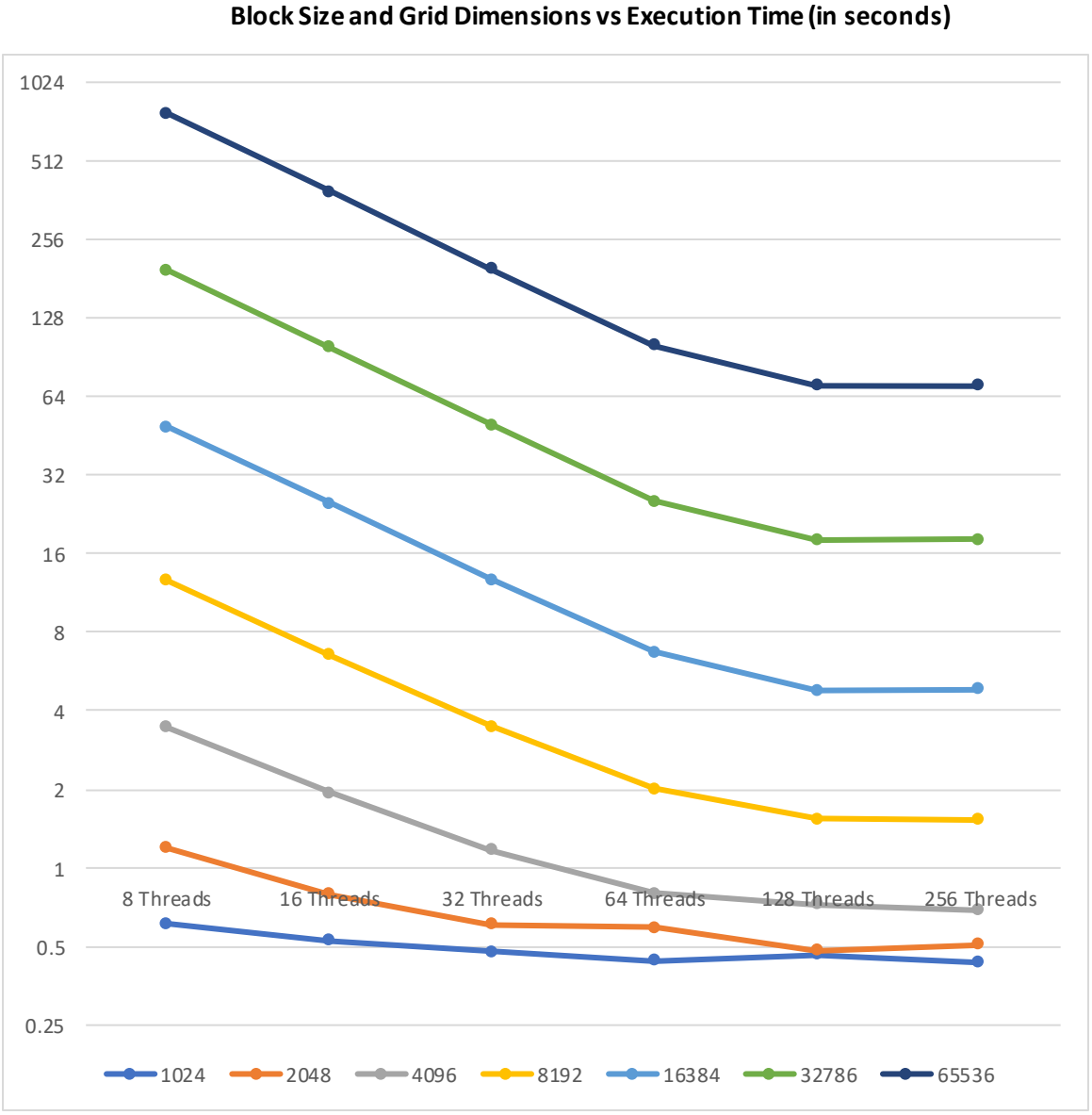
./highlife <pattern_number> <grid_size> <number_of_iterations> <block_size>

The program was run on AiMOS for 7 different grid sizes and 6 block sizes. The data from each run in plotted on the following two pages. The execution time grows exponentially with grid size. Since increasing grid size increases the number of cell updates per second quadratically, this is expected. For each grid size, the slowest execution was with a block size of 8 threads. This is expected since this configuration yields the highest number of blocks. Since each SM can only run one block at a time, such a small block will drastically decrease performance. With a hardware warp size of 32 threads, any block size less than 32 threads should always run slower than a block size of 32 or above.

It was expected the speed would peak at 32 threads and remain relatively constant from 32 threads and up. In reality, most grid sizes saw speed improvements up to 128 threads, with 256 running at a similar speed. This could be because of optimizations the hardware made to run one block on multiple SM's concurrently. This optimization effectively allowed the hardware to simulate running with a warp size of 128 threads. In other tests on a smaller system, there was a clear flatlining of performance from 32 threads and above, the expected result. The AiMOS system performed differently to that smaller system.

The second figure lists all run in order of the greatest number of cell updates per second. The greatest speed was achieved by the grid size of 65536 x 65536 and a block size of 128.

Highlife Assignment 2



All 42 Simulations Ordered by Cell Updates per Second

$$Cell\ Updates\ /\ sec\ = \frac{(gridsize)^2 * 1024}{executiontime}$$

Grid Size	Thread Count	Execution Time (s)	Cell Updates / sec
65536	128 Threads	70.352	62,514,875,357
65536	256 Threads	70.647	62,253,832,592
32786	128 Threads	18.038	61,022,281,800
32786	256 Threads	18.12	60,746,132,401
16384	128 Threads	4.786	57,433,745,705
16384	256 Threads	4.822	57,004,957,890
8196	256 Threads	1.526	45,076,410,212
8196	128 Threads	1.546	44,493,274,246
65536	64 Threads	100.218	43,884,796,255
32786	64 Threads	25.408	43,321,785,229
16384	64 Threads	6.698	41,038,803,664
8196	64 Threads	2.006	34,290,429,703
4096	256 Threads	0.688	24,970,740,093
4096	128 Threads	0.722	23,794,832,665
65536	32 Threads	197.239	22,298,057,236
32786	32 Threads	49.738	22,130,361,476
16384	32 Threads	12.705	21,635,411,802
4096	64 Threads	0.801	21,448,026,447
8196	32 Threads	3.489	19,715,277,152
4096	32 Threads	1.17	14,683,648,875
65536	16 Threads	391.147	11,243,973,522
32786	16 Threads	98.383	11,188,110,945
16384	16 Threads	24.883	11,046,815,374
8196	16 Threads	6.566	10,476,180,625
2048	128 Threads	0.481	8,929,245,938
4096	16 Threads	1.939	8,860,169,770
2048	256 Threads	0.507	8,471,335,890
2048	64 Threads	0.589	7,291,964,849
2048	32 Threads	0.605	7,099,119,498
65536	8 Threads	779.602	5,641,399,728
32786	8 Threads	195.384	5,633,623,629
16384	8 Threads	49.105	5,597,758,007
8196	8 Threads	12.622	5,449,738,709
2048	16 Threads	0.79	5,436,667,463
4096	8 Threads	3.453	4,975,345,840
2048	8 Threads	1.192	3,603,160,483
1024	256 Threads	0.434	2,474,059,502
1024	64 Threads	0.441	2,434,788,717
1024	128 Threads	0.466	2,304,167,004
1024	32 Threads	0.479	2,241,632,200
1024	16 Threads	0.526	2,041,334,266
1024	8 Threads	0.612	1,754,480,105