GPUs Hands On 2
May 11, 2022

J L Gouws
19G4436

Each kernel runs 1000 times; Each kernel runs exclusively on the device. The program finds the average run time.

2D Block	**Step 3**		**Step 4**		**Step 5**	
	2D Grid Size	Execution Time ms	1D Grid size 1 datum per thread	Execution time ms	Grid Size 16 data per thread	Execution time ms
32×32	128×128	2.725741	16384	2.691122	1024	2.777440
32×16	128×256	2.694830	32768	2.704954	2048	2.790173
16×32	256×128	2.707887	32768	2.708225	2048	2.788703
16×16	256×256	2.686316	65536	2.711797	4096	2.792496
64×16	64×256	2.701534	16384	2.698435	1024	2.785574
16×64	256×64	2.825843	16384	2.696335	1024	2.788394
64×8	64×512	2.669709	32768	2.711425	2048	2.803214
8×64	512×64	3.444676	32768	2.717273	2048	2.790220
64×2	64×2048	2.715393	131072	2.725050	8192	2.825520
2×64	2048×64	5.792675	131072	2.726640	8192	2.835794

Table 1: Average execution times of Kernel

Observations:
On average, the performance seems to be worse with larger block size. Larger block sizes result in fewer blocks being launched. Fewer blocks being launched can result in unoccupied streaming mulitprocessors. Less usage of the streaming mulitprocessors results in fewer FLOPs–that is lower performance.

When each thread does 16 data items per thread, the threads must stride accross the data in order to achieve decent performance. If the threads do not stride over the data, the kernel performs poorly. This performance reduction is due to memory access in the GPU. I think that striding through the data allows the kernel to use the spatial locality of cache. This use of the cache's spatial locality increases performance, because new data items do not have to be fetched from global memory for every execution of one or two warps.

With striding, all the kernels seem to perform similarly. The normal one dimensional grid seems to perform the best on average. The one dimensional grid with 16 data per thread seems to perform the worst on average. The two dimensional grid has the most variation with a few outliers.

The execution times are not symmetric with block sizes. The $2D$ grid kernel performs fine with a 64×2 block size, but poorly with a 2×64 block size. This might be attributed to warp divergence. I should use the profiler to confirm this.