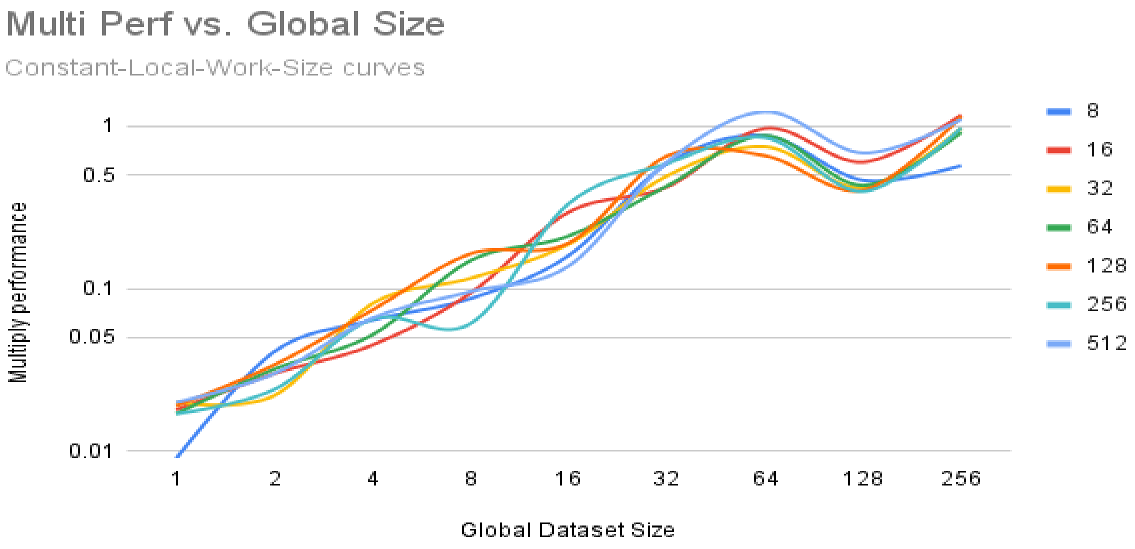
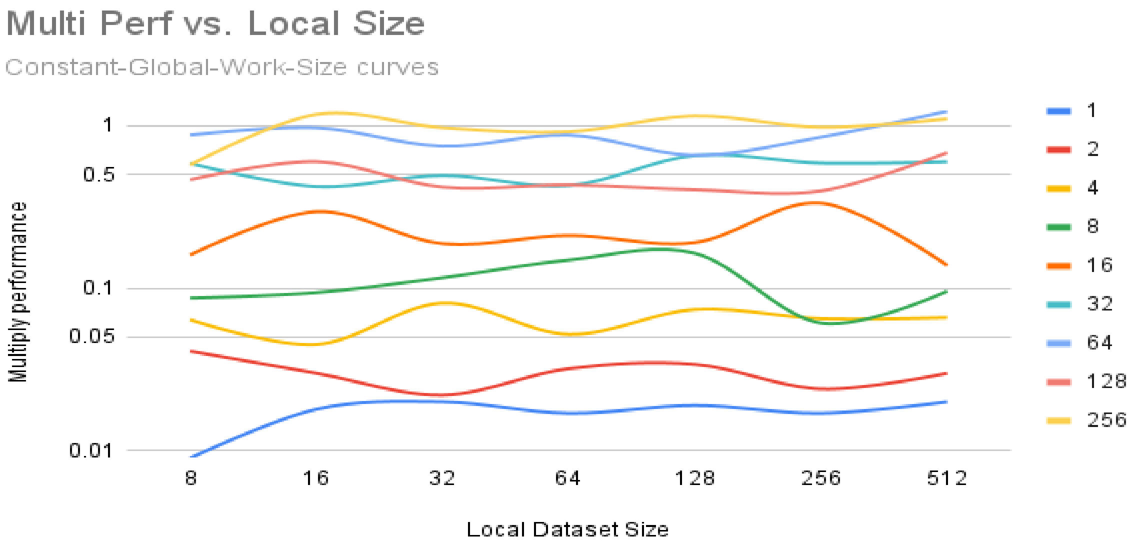
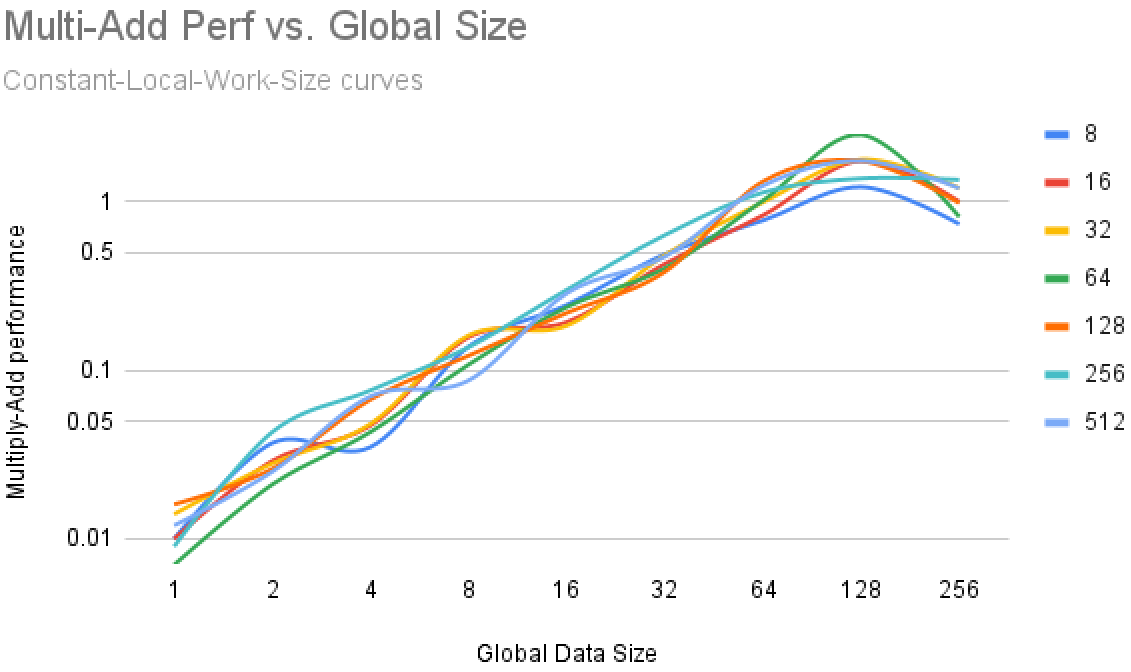
Adrien Protzel

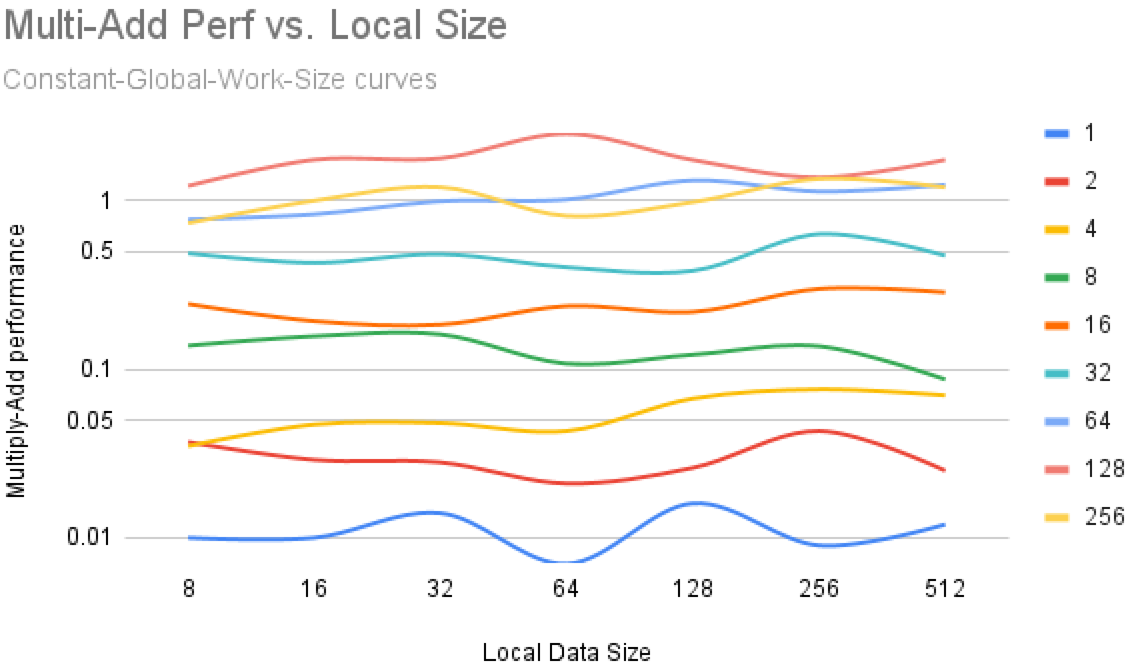
1. Home desktop SSH onto rabbit server
2. (Shown Below)
3. The graphs seem sort of erratic although you can see there is some plateauing in using the Local size vs. the Global size that has an exponential increase.
4. I believe this is because of the amount of data that is being worked on. The global size jumps by large amounts while the local sizes have smaller steps which limit the amount of performance that can be achieved.
5. From the data I received, they look pretty similar, although the -Add is a bit more erratic and varied (this could be due to rabbit server lag).
6. Based on this data, the GPU is able to split into work groups and do the math and send back a result reliably.






Multi-Add Perf vs. Global Size

Constant-Local-Work-Size curves


Multi-Add Perf vs. Local Size

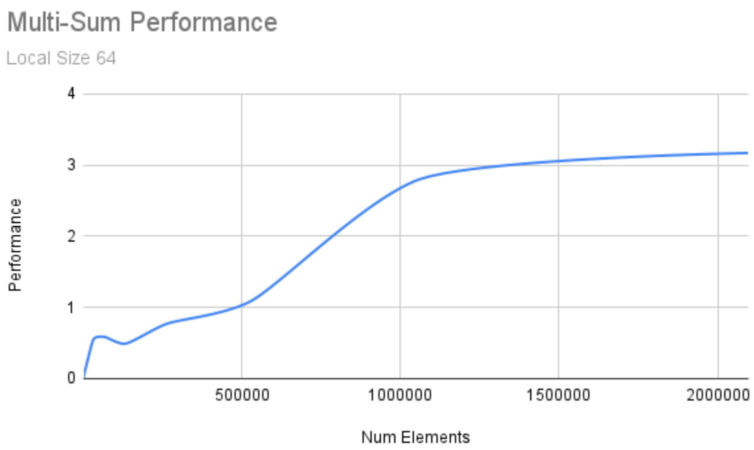
Constant-Global-Work-Size curves

Multi					Multi+Add			
Global	Local	Work Groups	Performance		Global	Local	Work Groups	Performance
1	8	128	0.009		1	8	128	0.01
2	8	256	0.041		2	8	256	0.037
4	8	512	0.064		4	8	512	0.035
8	8	1024	0.087		8	8	1024	0.138
16	8	2048	0.16		16	8	2048	0.244
32	8	4096	0.587		32	8	4096	0.488
64	8	8192	0.876		64	8	8192	0.775
128	8	16384	0.465		128	8	16384	1.224
256	8	32768	0.573		256	8	32768	0.737
1	16	64	0.018		1	16	64	0.01
2	16	128	0.03		2	16	128	0.029
4	16	256	0.045		4	16	256	0.047
8	16	512	0.094		8	16	512	0.157
16	16	1024	0.296		16	16	1024	0.193
32	16	2048	0.422		32	16	2048	0.428
64	16	4096	0.971		64	16	4096	0.831
128	16	8192	0.601		128	16	8192	1.751
256	16	16384	1.173		256	16	16384	1.001
1	32	32	0.02		1	32	32	0.014
2	32	64	0.022		2	32	64	0.028
4	32	128	0.081		4	32	128	0.048
8	32	256	0.116		8	32	256	0.161
16	32	512	0.188		16	32	512	0.184
32	32	1024	0.494		32	32	1024	0.482
64	32	2048	0.748		64	32	2048	0.993
128	32	4096	0.419		128	32	4096	1.785
256	32	8192	0.972		256	32	8192	1.204
1	64	16	0.017		1	64	16	0.007
2	64	32	0.032		2	64	32	0.021
4	64	64	0.052		4	64	64	0.043
8	64	128	0.149		8	64	128	0.108
16	64	256	0.211		16	64	256	0.237

32	64	512	0.428		32	64	512	0.403
64	64	1024	0.876		64	64	1024	1.017
128	64	2048	0.432		128	64	2048	2.496
256	64	4096	0.918		256	64	4096	0.813
1	128	8	0.019		1	128	8	0.016
2	128	16	0.034		2	128	16	0.026
4	128	32	0.074		4	128	32	0.067
8	128	64	0.164		8	128	64	0.122
16	128	128	0.191		16	128	128	0.219
32	128	256	0.655		32	128	256	0.384
64	128	512	0.658		64	128	512	1.319
128	128	1024	0.404		128	128	1024	1.746
256	128	2048	1.149		256	128	2048	0.982
1	256	4	0.017		1	256	4	0.009
2	256	8	0.024		2	256	8	0.043
4	256	16	0.065		4	256	16	0.076
8	256	32	0.061		8	256	32	0.137
16	256	64	0.334		16	256	64	0.3
32	256	128	0.589		32	256	128	0.634
64	256	256	0.851		64	256	256	1.135
128	256	512	0.397		128	256	512	1.375
256	256	1024	0.979		256	256	1024	1.35
1	512	2	0.02		1	512	2	0.012
2	512	4	0.03		2	512	4	0.025
4	512	8	0.066		4	512	8	0.07
8	512	16	0.096		8	512	16	0.087
16	512	32	0.138		16	512	32	0.286
32	512	64	0.6		32	512	64	0.472
64	512	128	1.227		64	512	128	1.244
128	512	256	0.683		128	512	256	1.749
256	512	512	1.103		256	512	512	1.201

1. (Shown Below)
2. It plateaus.
3. In general, the performance should be much better, especially the larger your numbers are supposed to compute. Because you aren't checking the arrays much afterwards.
4. Without needing to double check each time line in multiplication, the summation is faster on the GPU.

Multi+Sum		
Local	Num Elements	Performance
32	1024	0.009
32	2048	0.02
32	4096	0.038
32	8192	0.11
32	16384	0.187
32	32768	0.38
32	65536	0.739
32	131072	0.288
32	262144	0.95
32	524288	1.748
32	1048576	1.925
32	2097152	0.253
64	1024	0.009
64	2048	0.023
64	4096	0.075
64	8192	0.146
64	16384	0.292
64	32768	0.551
64	65536	0.588
64	131072	0.49
64	262144	0.768
64	524288	1.078
64	1048576	2.782
64	2097152	3.171
128	1024	0.017
128	2048	0.035



128	4096	0.043
128	8192	0.136
128	16384	0.284
128	32768	0.359
128	65536	0.671
128	131072	0.599
128	262144	1.077
128	524288	1.666
128	1048576	3.225
128	2097152	4.718
256	1024	0.014
256	2048	0.039
256	4096	0.05
256	8192	0.143
256	16384	0.258
256	32768	0.546
256	65536	0.551
256	131072	0.47
256	262144	1.173
256	524288	1.018
256	1048576	2.387
256	2097152	4.12