CS 475/575 -- Spring Quarter 2019

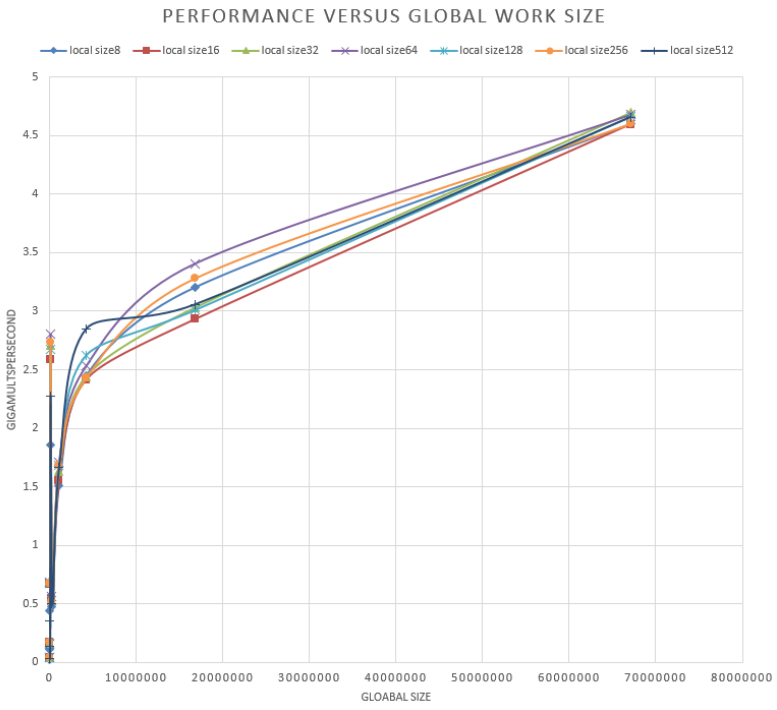
Project #5

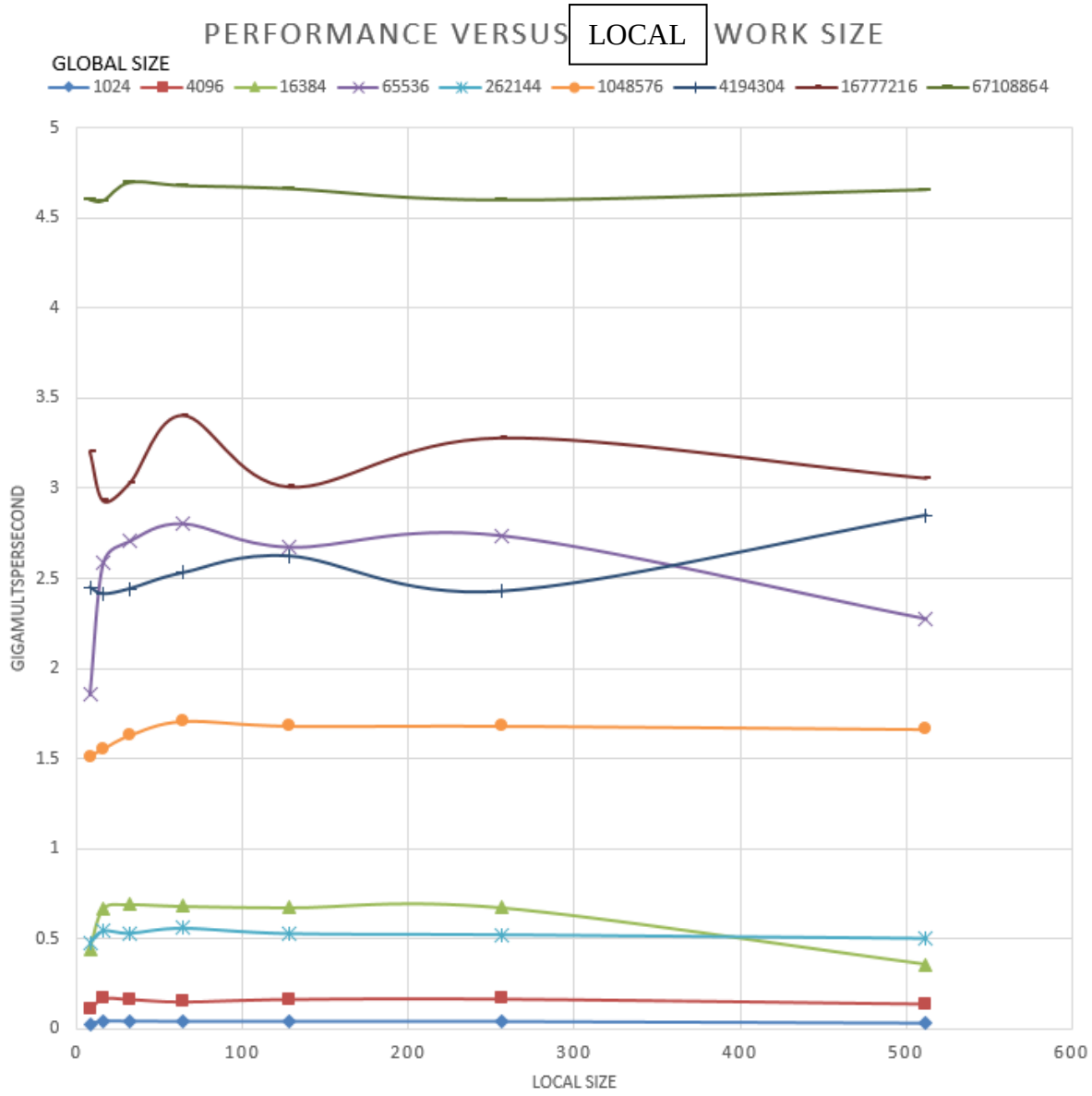
Jiawei Mo
moji@oregonstate.edu

running on gtx1070

Multiply table and graphs
Performance is GigaMultsPerSecond

gloabal size	local size8	local size16	local size32	local size64	local size128	local size256	local size512
1024	0.027	0.045	0.044	0.043	0.043	0.042	0.031
4096	0.109	0.171	0.167	0.153	0.167	0.169	0.141
16384	0.446	0.669	0.693	0.684	0.676	0.676	0.359
65536	1.855	2.584	2.706	2.804	2.674	2.738	2.277
262144	0.477	0.546	0.532	0.562	0.53	0.523	0.501
1048576	1.511	1.55	1.628	1.709	1.682	1.682	1.663
4194304	2.452	2.416	2.444	2.532	2.624	2.43	2.85
16777216	3.205	2.931	3.031	3.407	3.01	3.282	3.058
67108864	4.6	4.596	4.695	4.678	4.661	4.6	4.656

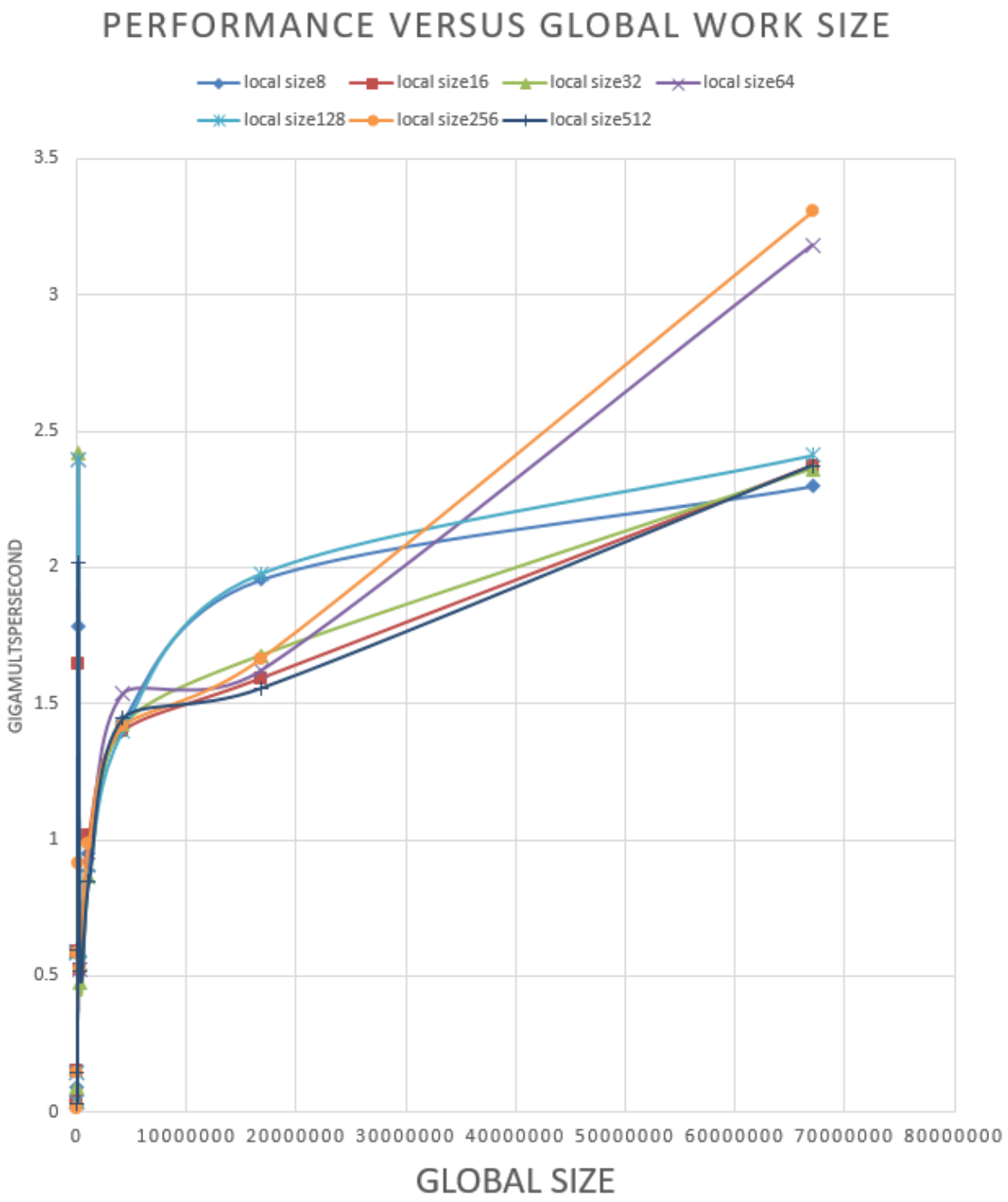


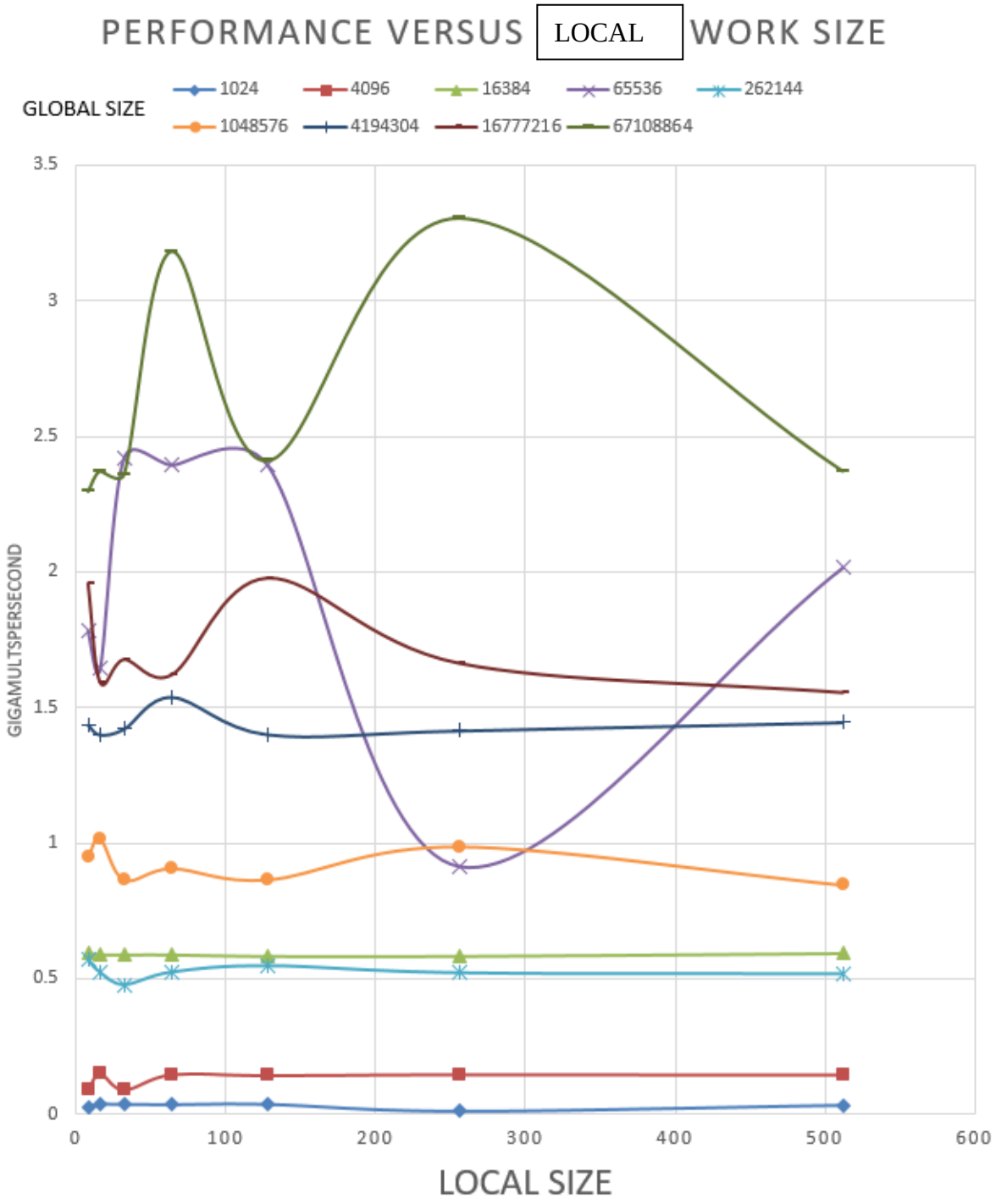


Multiply-Add table and graphs
Performance is GigaMultsPerSecond

gloabal size	local size8	local size16	local size32	local size64	local size128	local size256	local size512
1024	0.025	0.037	0.037	0.036	0.037	0.014	0.034
4096	0.092	0.151	0.092	0.148	0.145	0.148	0.147
16384	0.593	0.587	0.587	0.587	0.581	0.581	0.593
65536	1.783	1.643	2.421	2.396	2.395	0.916	2.017
262144	0.571	0.525	0.478	0.526	0.55	0.523	0.519
1048576	0.947	1.016	0.866	0.907	0.866	0.986	0.847
4194304	1.437	1.401	1.422	1.538	1.402	1.416	1.446

16777216	1.956	1.594	1.678	1.622	1.978	1.664	1.556
67108864	2.299	2.374	2.362	3.183	2.412	3.305	2.374





Commentary

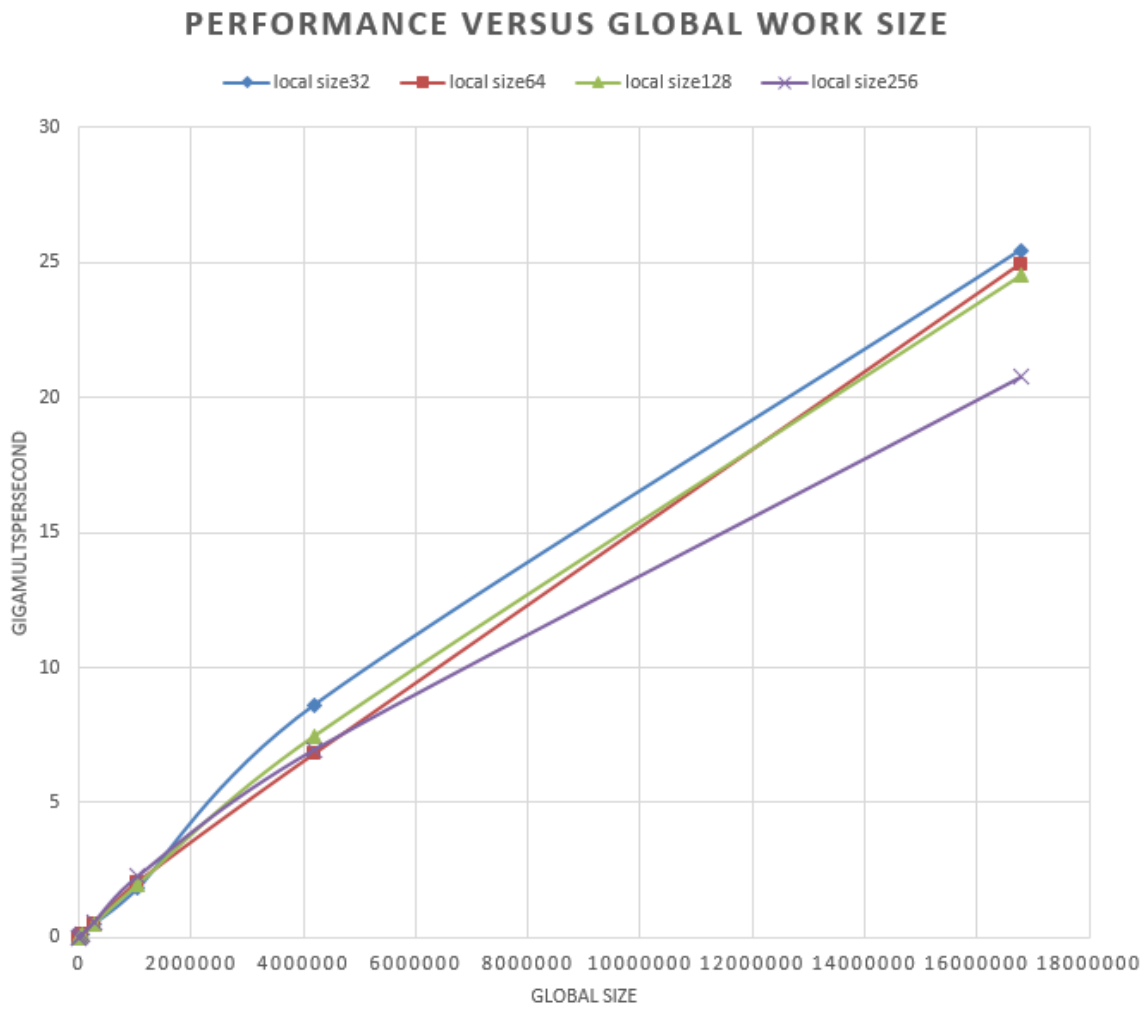
Both multiply and multiply-add have similar pattern.

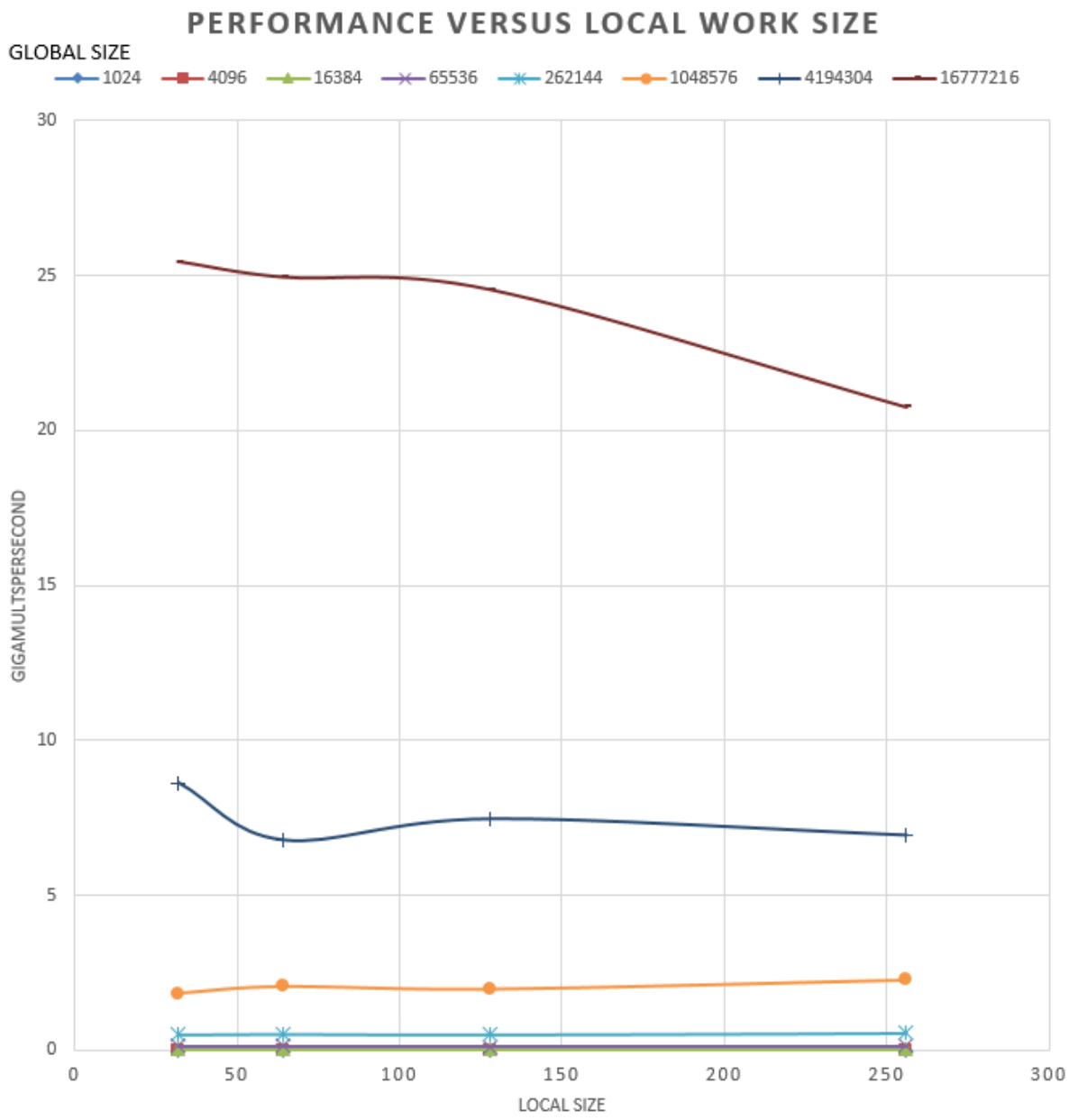
With performance vs global work size, the performance starts to drop after work size = 65536 and then rises again. After 70M work size, the performance gets a lot bigger. The GPU might be impacted by a relatively small work group size since there are not huge enough tasks to run on the GPU, which leads to a inconstantly increasing. As we see, the performance is keeping rising when it hits the max array size. Local size 64 and 256 shows a better performance than others, which implies that this GPU prefer those units size to process at a time.

With performance vs local work size, it indicates that most of their performance will be bouncing around a certain number. There might be other things that affects its performance. Anyway, the performance using GPU is extremely better than SIMD on CPU. Thanks to a number of GPU's computational units.

Reduction tables and graphs
Performance is GigaMultsPerSecond

gloabal size	local size32	local size64	local size128	local size256
1024	0.002	0.002	0.002	0.002
4096	0.008	0.009	0.008	0.009
16384	0.035	0.032	0.029	0.036
65536	0.13	0.134	0.106	0.142
262144	0.497	0.516	0.498	0.557
1048576	1.837	2.059	1.972	2.271
4194304	8.612	6.804	7.471	6.956
16777216	25.431	24.935	24.53	20.759





Commentary

As global size increasing, the performance is also increasing. And since the reduction method, it rockets up the performance up to 20G/s, which is the most efficient way to compute in GPU.