Project6

OpenCL Linear Regression

Name: Yi Chiun Chang
Email: changyic@oregonstate.edu

1. Tell what machine you ran this on

CyberDuck(rabbit.engr.oregonstate.edu & submit-a.engr.oregonstate.edu)

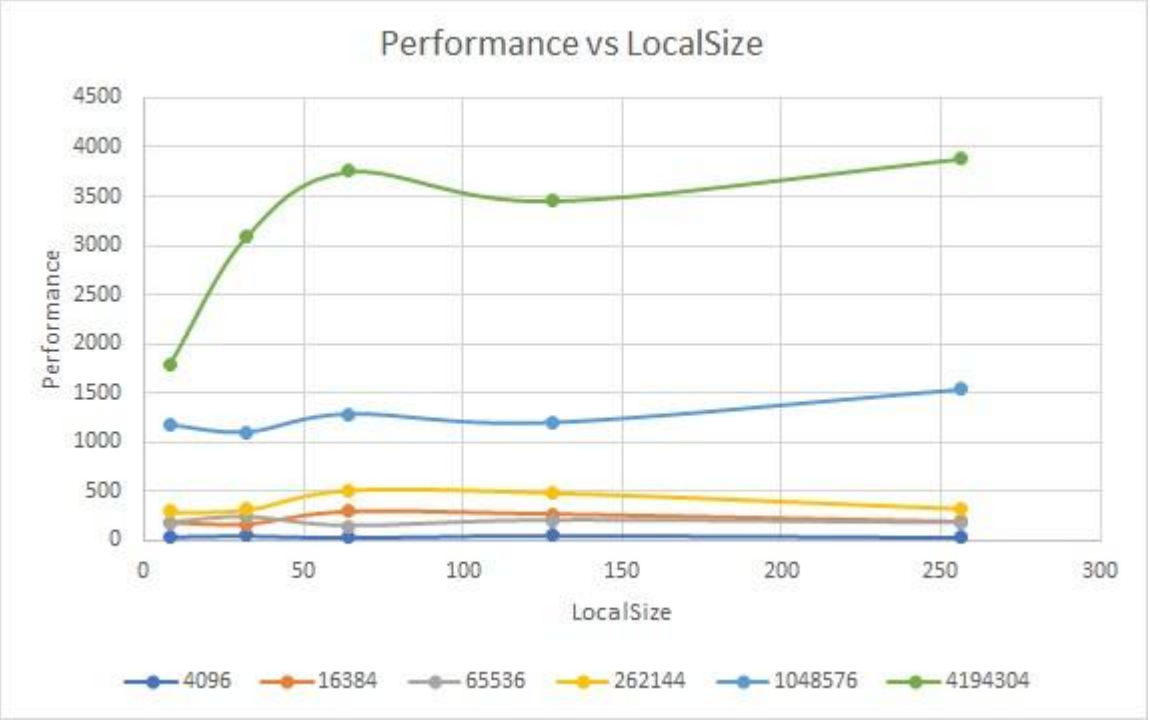
2. Show the table and graphs

Table:

I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 6.99 **	
4096	8	29.86
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 6.99 **	
4096	32	50.96
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 6.99 **	
4096	64	33.68
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 6.99 **	
4096	128	54.28
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 6.99 **	
4096	256	35.41
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
16384	8	182.61
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
16384	32	165.43
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
16384	64	295.12
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
16384	128	269.82
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
16384	256	191.58
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
65536	8	183.01
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
65536	32	242.74
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
65536	64	148.34
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
65536	128	205.41
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
65536	256	180.39

I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
262144	8	293.75
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
262144	32	316.64
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
262144	64	513.57
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
262144	128	490.69
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
262144	256	331.24
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
1048576	8	1178.87
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
1048576	32	1101.57
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
1048576	64	1289.24
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
1048576	128	1200.99
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.00	b = 7.00 **	
1048576	256	1539.32
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.01	b = 7.00 **	
4194304	8	1780.39
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.01	b = 7.00 **	
4194304	32	3087.32
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.01	b = 7.00 **	
4194304	64	3756.03
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.01	b = 7.00 **	
4194304	128	3458.83
I have selected Platform #0	Device #0: Vendor = NVIDIA	Type = CL_DEVICE_TYPE_GPU
** m = 5.01	b = 7.00 **	
4194304	256	3886.93

Graph:



3. What patterns are you seeing in the performance curves? What difference does the size of data make? What difference does the size of each work-group make?

Firstly, in the "Performance vs DataSize" chart, we observe that as the data size (DataSize) increases, the performance improves across all work-group sizes (LocalSize). For smaller data sizes, the performance increases rapidly, but as the data size continues to grow, the rate of performance increase tends to stabilize. Larger data sizes show more consistent performance improvements, indicating that

the GPU's parallel processing capabilities are being fully utilized. Specifically, a larger work-group size (such as 256) performs best with larger data sizes, while smaller work-group sizes (such as 8) show lower performance. Secondly, in the "Performance vs LocalSize" chart, the performance tends to increase with the work-group size up to a certain point, usually around 64-128, and then stabilizes or slightly decreases. This suggests that mid-range work-group sizes can balance workload distribution and overhead, achieving optimal performance. Too small or too large work-group sizes can lead to suboptimal performance due to either underutilization or excessive overhead.

In summary, larger data sizes generally yield better performance because they more effectively utilize the GPU's parallel processing capabilities. The optimal work-group size is typically around 64-128, maximizing the GPU's efficiency. Both data size and work-group size need to be carefully tuned to achieve the best performance in GPU-based linear regression tasks.

4. Why do you think the patterns look this way?

The observed patterns in the performance curves can be attributed to the way GPUs handle parallel processing and workload distribution. Initially, as the data size (DataSize) increases, the performance improves rapidly across all work-group sizes (LocalSize) because more of the GPU's cores are engaged, leading to efficient parallel processing. However, as the data size continues to grow, the rate of performance increase stabilizes since the GPU's cores are already fully utilized, resulting in consistent performance gains. This is because larger datasets amortize the overhead of managing parallel tasks over a greater number of operations, ensuring efficient resource utilization. In the "Performance vs LocalSize" chart, mid-range work-group sizes (64-128) show optimal performance as they balance efficient workload distribution and minimal overhead. Smaller work-group sizes underutilize the GPU's capabilities, while larger sizes introduce significant overhead, both leading to suboptimal performance. Thus, the best performance is achieved by tuning both data size and work-group size, ensuring maximum GPU efficiency and minimal overhead, which explains the observed performance trends