First Home work report:

**The results of residual and MFLOPs for Vector Norm in BATTLECAT2 cluster:**

I have generated a random number between 0 and 1 as each component of the vector x with MATLAB. I have then calculated the norm of that vector with command norm(x,2) in MATLAB(I have attached them to folder). Then I just calculated Vector Norm by my C code. It should be noted that I used double precision variables in order to get the precision of 1x10-15. The dimension of vector changes between 10 to 1000 as requested in HW. Also, in another method I have included a section to allocate memory in buffer for the input file.

*The results without memory allocation (malloc(..)):(This is the method I use for comparing the results of Vector Norm operation with the other two operations)*

The Residual: r = abs(my Norm- MATLAB Norm) / (my Norm \* EPS (~1x10-15))



As it should be in order 1, so it seems that the code has sufficient accuracy.

The results of MFLOPS/s are as follows. I have divided the number of operations per time of calculation. The rate of execution have tendency to become a plateau in larger n.



The Ceiling theoretical peak performance may be calculated as follows,

(2.13 GCycle/s\*(4Flops/cycle or 2Flops/cycle)\*1core = 8.52GFlops/s or 4.26GFlops/s). So, the peak performance is either 10 or 20 times larger than maximum reachable rate of execution in this range of Matrix dimension. This huge difference between these two might be attributed to: 1: The efficiency of the program that is written by me compared to most efficient way of writing it. 2: The restriction of memory access (Latency and ..), so some instructions may be required to access the data of Matrices and vectors. 3: The role of operating system and all services that is available at the time of running may not be ignored.

*The results with considering memory allocation:*

The residual does not change very much. However, interestingly the rate of execution decreases noticeably. The probably the default allocation of memory by compiler is more efficient.



**The results of residual and MFLOPs for Matrix Vector Multiplication in BATTLECAT2 cluster:**

I have defined 50 different dimensions from 10 to 1000. As it was done for Vector norm operation, I have generated a random number in long format in MATLAB as each component of matrix and vectors. Then, I have done the same multiplication in MATLAB and I compared the results by defining resuidual (r = Norm(y-) / Norm(y)\*eps). The results are as follows:

The residual is again in the order of 1 which reflects that the results are accurate enough.

(Time Analysis): For MFLOPs/s, it's obvious that other than the first sudden changes, the rate of execution remains the same for different matrix dimension. The overall rate of execution is also decreased compared to vector norm operation. This tells me that although the number of operation is increased by factor n, the time of execution is increased by slightly larger factor, let's consider n = 200 then based on figures : (325\* = 160 x = 406) so time has roughly increased by factor 406 in that dimension. Also, remaining unchanged in almost all dimensions tells me that the time increment has functionality of n2. So, this is probably the reason that this graph is steady and Vector norm graph has increasing behavior. One more thing to state is that, the necessary amount of data that is needed for Vector Matrix operation is n2, while that of Vector norm is n. So the Latency and Network Bandwidth are some factors that may play a role here. This will surely results in decreasing the rate of execution.

**The results of residual and MFLOPs for Matrix Matrix Multiplication in BATTLECAT2 cluster:**



I couldn't get to higher dimensions which I think was because of storage restriction, because I used double precision in my calculations. Based on the discussion above in Time analysis, I expected that the behavior would be decreasing. Because the time has n2 functionality while the operations are increasing with power of three. I could go to higher dimensions in Newton cluster and it confirmed that the rate of execution decreases with increasing the dimension of the matrix.

**Matrix Matrix Multiplication in Newton cluster:**



**Matrix Matrix Multiplication in hydra cluster:**

I changed the double precision to float, so I could go to higher dimensions in hydra.

****

**Vector Matrix Multiplication in hydra cluster:**

****

The Ceiling theoretical peak performance may be calculated as follows,

(2.66 GCycle/s\*(4Flops/cycle or 2Flops/cycle)\*1core = 10.64 GFlops/s or 5.32 GFlops/s).

the maximum obtainable MFlops/s is higher for hydra as they have faster Processor. However, the difference between the maximum value and peak performance is higher than 10 and 20 for these machines. It might be attributed to Memory Latency.