Application Performance on P100-PCIe GPUs

Authors: Rengan Xu, Frank Han and Nishanth Dandapanthu. Dell EMC HPC Innovation Lab. Feb 2017

Introduction to P100-PCIe GPU

This blog describes the performance analysis on NVIDIA® Tesla® P100™ GPUs on a cluster of Dell PowerEdge C4130 servers. There are two types of P100 GPUs: PCIe-based and SXM2-based. In PCIe-based server, GPUs are connected by PCIe buses and one P100 delivers around 4.7 and 9.3 TeraFLOPS of double and single precision performance, respectively. And in P100-SXM2, GPUs are connected by NVLink and one P100 delivers around 5.3 and 10.6 TeraFLOPS of double and single precision performance, respectively. This blog focuses on P100 for PCIe-based servers, i.e. P100-PCIe. We have already analyzed the P100 performance for several deep learning frameworks in this blog. The objective of this blog is to compare the performance of HPL, LAMMPS, NAMD, GROMACS, HOOMD-BLUE, Amber, ANSYS Mechanical and RELION. The hardware configuration of the cluster is the same as in the deep learning blog. Briefly speaking, we used a cluster of four C4130 nodes, each node has dual Intel Xeon E5-2690 v4 CPUs and four NVIDIA P100-PCIe GPUs and all nodes are connected with EDR Infiniband. Table 1 shows the detailed information about the hardware and software used in every compute node.

Table 1: Experiment Platform and Software Details

Platform	PowerEdge C4130 (configuration G)
Processor	2 x Intel Xeon CPU E5-2690 v4 @2.6GHz (Broadwell)
Memory	256GB DDR4 @ 2400MHz
Disk	9TB HDD
GPU	P100-PCIe with 16GB GPU memory
Nodes Interconnects	Mellanox ConnectX-4 VPI (EDR 100Gb/s Infiniband)
Infiniband Switch	Mellanox SB7890
Software and Firmware	
Operating System	RHEL 7.2 x86_64
Linux Kernel Version	3.10.0-327.el7
BIOS	Version 2.3.3
CUDA version and driver	CUDA 8.0.44 (375.20)
OpenMPI compiler	Version 2.0.1

GCC compiler	4.8.5
Intel Compiler	Version 2017.0.098
Applications	
HPL	Version hpl_cuda_8_ompi165_gcc_485_pascal_v1
LAMMPS	Version Lammps-30Sep16
NAMD	Version NAMD_2.12_Source
GROMACS	Version 2016.1
HOOMD-blue	Version 2.1.2
Amber	Version 16update7
ANSYS Mechanical	Version 17.0
RELION	Version 2.0.3

High Performance Linpack (HPL)

HPL is a multicomputer parallel application to measure how fast computers solve a dense n by n system of linear equations using LU decomposition with partial row pivoting and designed to be run at very large scale. The HPL running on the experimented cluster uses the double precision floating point operations. Figure 1 shows the HPL performance on the tested P100-PCIe cluster. It can be seen that 1 P100 is 3.6x faster than 2 x E5-2690 v4 CPUs. HPL also scales very well with more GPUs within nodes or across nodes. Recall that 4 P100 is within a server and therefore 8, 12 and 16 P100 are in 2, 3 and 4 servers. 16 P100 GPUs has the speedup of 14.9x compared to 1 P100. Note that the overall efficiency is calculated as: HPL Efficiency = rMax / (CPUs rPeak + GPUs rPeak), where rPeak is the highest theoretical FLOPS result that could be achieved with base clock, and the number reported by HPL is rMax and is the real performance that can be achieved. HPL cannot be run at the max boost clock. It is typically run at some number in between but the average is close to the base clock then to the max boost clock. That is why we used base clock for rPeak calculation. Although we also included CPU rPeak in the efficiency calculation, when running HPL on P100 we set DGEMM_SPLIT=1.0 which means CPU is not really contributing to the DGEMM, but only handling other overhead so it is not actually contributing a lot of FLOPS. Although we observed that CPUs stayed fully utilized they were just handling the overhead and data movement to keep the GPUs fed. What is the most important for P100 is rMax is really big.

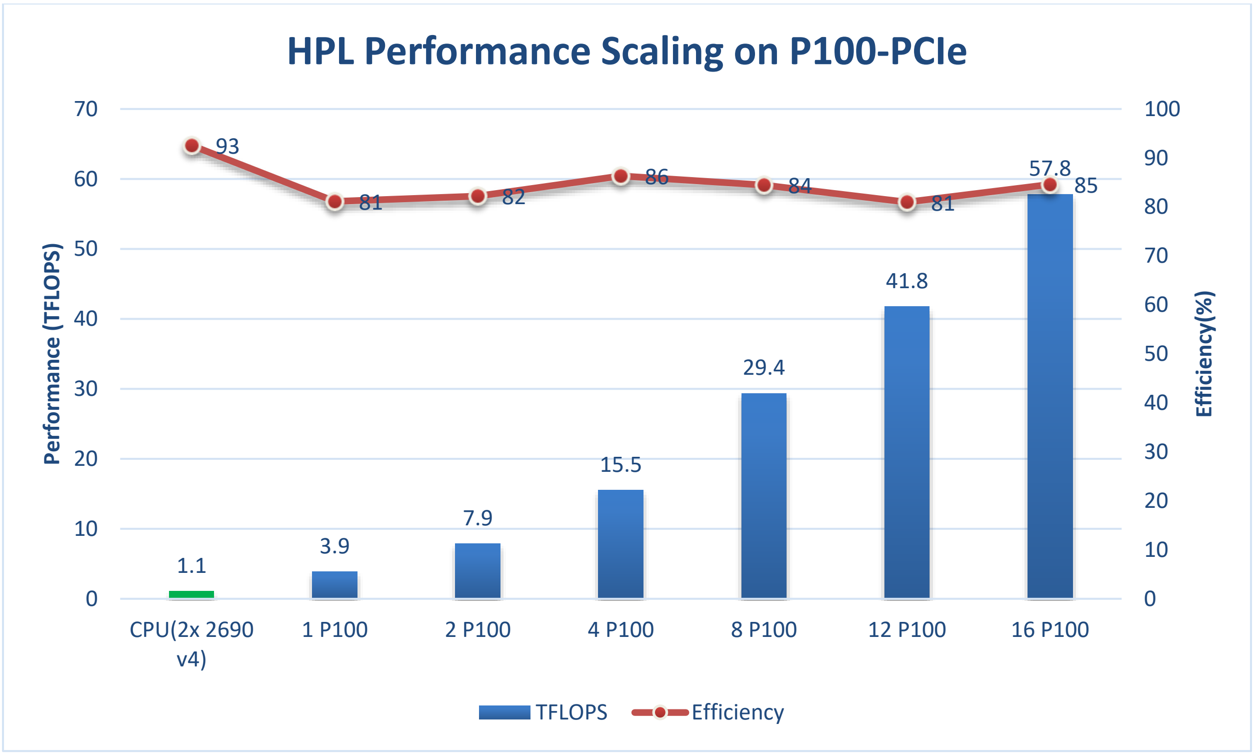

Figure 1: HPL performance on P100-PCIe

NAMD

NAMD (for NAnoscale Molecular Dynamics) is a molecular dynamics application designed for high-performance simulation of large biomolecular systems. The dataset we used is Satellite Tobacco Mosaic Virus (STMV) which is a small, icosahedral plant virus that worsens the symptoms of infection by Tobacco Mosaic Virus (TMV). This dataset has 1,066,628 atoms and it is the largest dataset on NAMD utilities website. The performance metric in the output log of this application is "days/ns" (the lower the better). But its inverted metric "ns/day" is used in our plot since that is what most molecular dynamics users focus on. The average of all occurrences of this value in the output log was used. Figure 2 shows the performance within 1 node. It can be seen that the performance of using 2 P100 is better than that of using 4 P100. This is probably because of the communications among different CPU threads. This application launches a set of workers threads that handle the computation and communication threads that handle the data communication. As more GPUs are used, more communication threads are used and more synchronization is needed. In addition, based on the profiling result from NVIDIA's CUDA profiler called nvprof, with 1 P100 the GPU computation takes less than 50% of the whole application time. According to Amdahl's law, the speedup with more GPUs will be limited by another 50% work that is not parallelized by GPU. Based on this observation, we further ran this application on multiple nodes with two different settings (2 GPUs/node and 4 GPUs/node) and the result is shown in Figure 3. The result shows that no matter how many nodes are used, the performance of 2 GPUs/node is always better than 4 GPUs/node. Within a node, 2 P100 GPUs is 9.5x faster than dual CPUs.

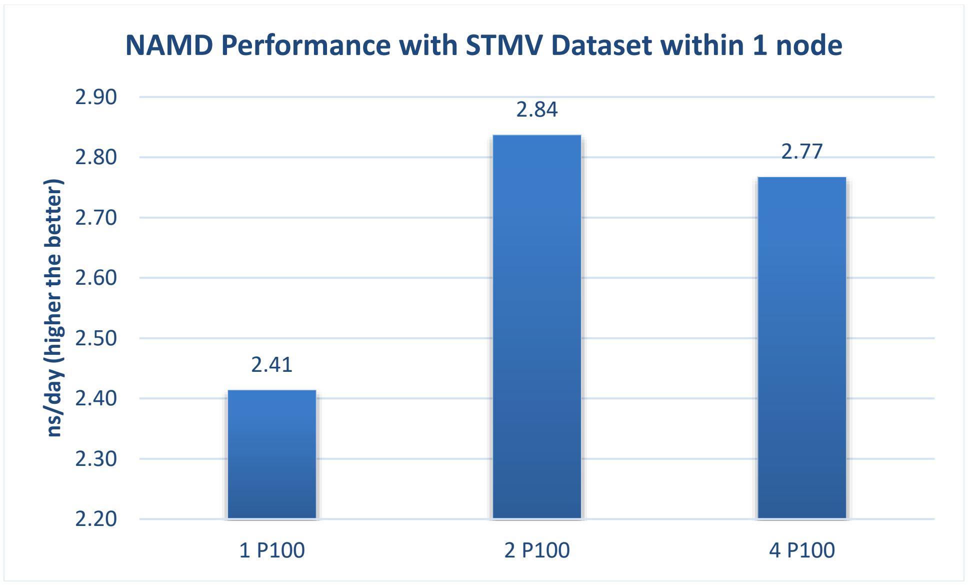

Figure 2: NAMD Performance within 1 P100-PCIe node

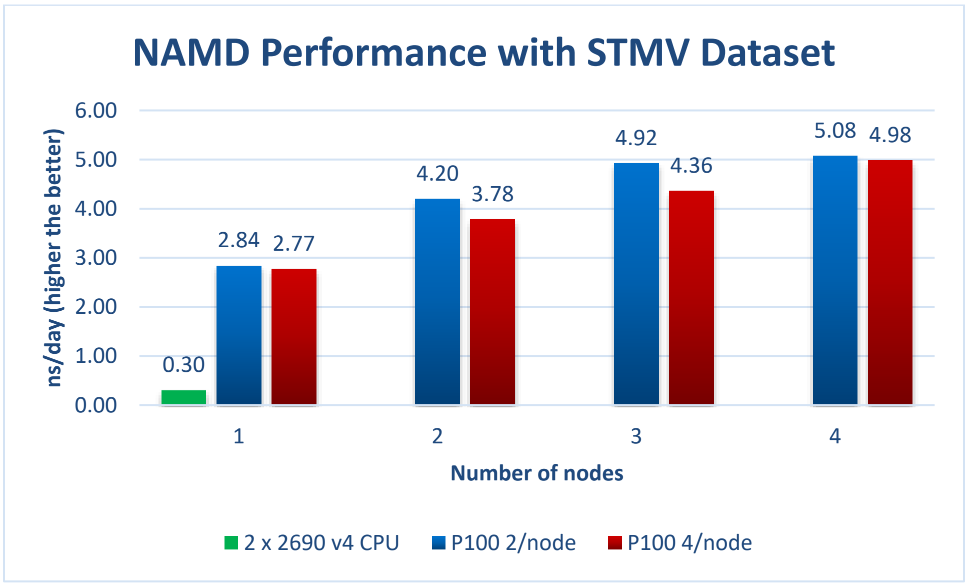

Figure 3: NAMD Performance across Nodes

GROMACS

GROMACS (for GROningen MAchine for Chemical Simulations) primarily does simulations for biochemical molecules (bonded interactions). But because of its efficiency in calculating non-bonded interactions (atoms not linked by covalent bonds), the user base is expanding to non-biological systems. Figure 4 shows the performance of GROMACS on CPU, K80 GPUs and P100-PCIe GPUs. Since one K80 has two internal GPUs, from now on when we mention one K80 it always refers to two internal GPUs instead of one of the two internal GPUs. When testing with K80 GPUs, the same P100-PCIe GPUs based servers were used. Therefore, the CPUs and memory were kept the same and the only difference is that P100-PCIe GPUs were replaced to K80 GPUs. In all tests, there were four GPUs per server and all GPUs were utilized. For example, the 3 node data point is with 3 servers and 12 total GPUs. The performance of P100-PCIe is 4.2x – 2.8x faster than CPU from 1 node to 4 nodes, and is 1.5x – 1.1x faster than K80 GPU from 1 node to 4 nodes.

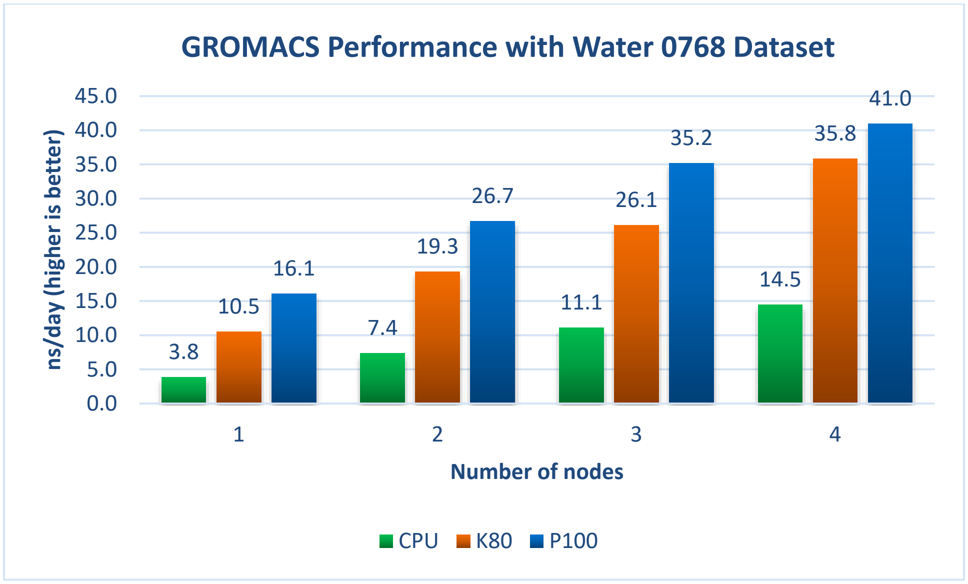

Figure 4: GROMACS Performance on P100-PCIe

LAMMPS

LAMMPS (for Large Scale Atomic/Molecular Massively Parallel Simulator) is a classic molecular dynamics code, capable of simulations for solid-state materials (metals, semi-conductors), soft matter (biomolecules, polymers) and coarse-grained or mesoscopic systems. It can be used to model atoms or more generically as a parallel particle simulator at the atomic, meso or continuum scale. The dataset we used was LJ (Lennard-Jones liquid benchmark) which contains 512000 atoms. There are two GPU implementations in LAMMPS: GPU library version and kokkos version. In the experiment, we used kokkos version since it was much faster than the GPU library version.

Figure 5 shows LAMMPS performance on CPU and P100-PCIe GPUs. Using 16 P100 GPUs is 5.8x faster than using 1 P100. The reason that this application did not scale linearly is that the data transfer (CPU->GPU, GPU->CPU and GPU->GPU) time increases as more GPUs are used although the computation part reduces linearly. And the reason that the data transfer time increases is because this application requires the data communication among all GPUs used. However, the configuration G we used only allows Peer-to-Peer (P2P) access for two pairs of GPUs: GPU 1 - GPU 2 and GPU 3 - GPU 4. GPU 1/2 cannot communicate with GPU 3/4 directly. If the communication is needed, the data must go through CPU which slows the communication. The configuration B is able to ease this issue as it allows P2P access among all four GPUs within a node. The comparison between configuration G and configuration B is shown in Figure 6. By running LAMMPS on a configuration B server with 4 P100, the performance metric “timesteps/s” was improved to 510 compared to 505 in configuration G, resulting in 1% improvement. The reason why the improvement is not significant is because the data communication takes only less than 8% of the whole application time when running on configuration G with 4 P100. Figure 7 also compared the performance of P100-PCIe with that of CPU and K80 GPUs for this application. It is shown that within 1 node, 4 P100-PCIe is 6.6x faster than 2 E5-2690 v4 CPUs and 1.4x faster than 4 K80 GPUs.

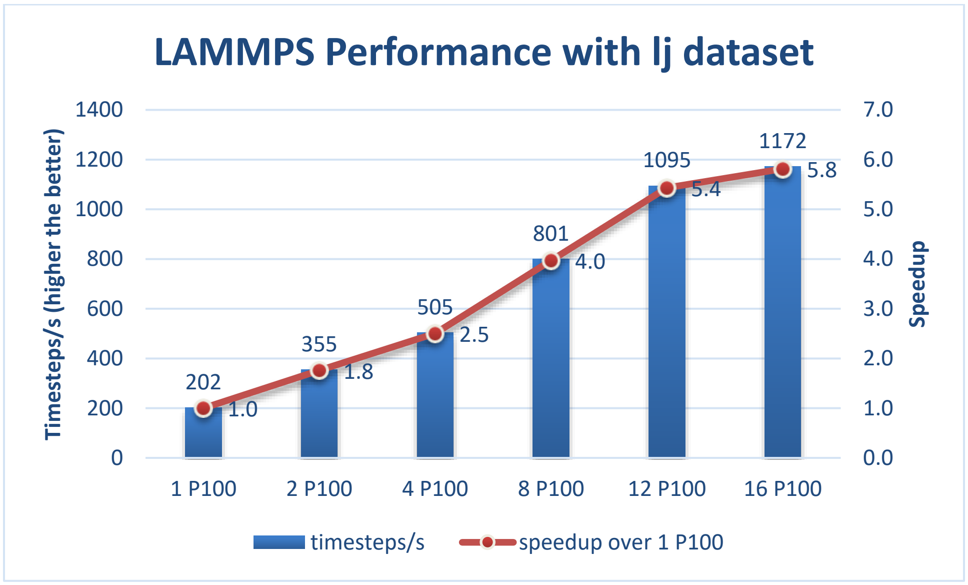

Figure 5: LAMMPS Performance on P100-PCIe

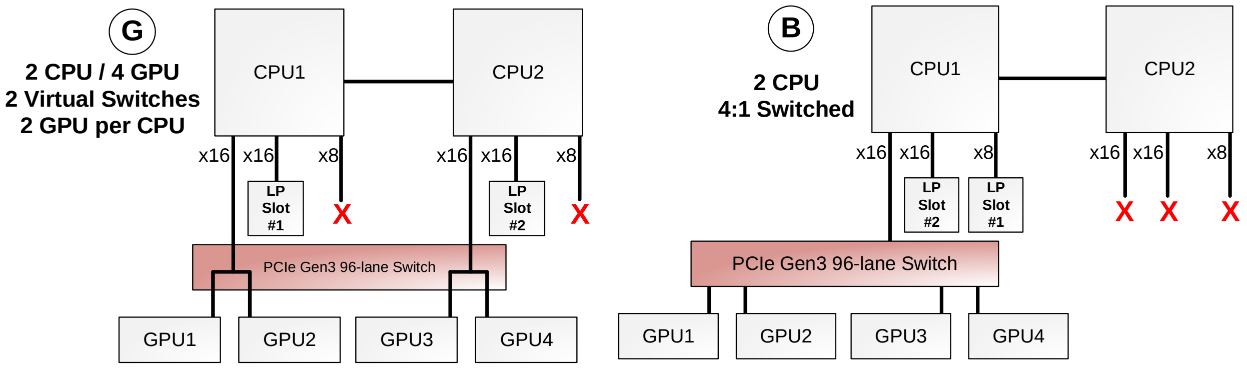

Figure 6 : Comparison between Configuration G and Configuration B

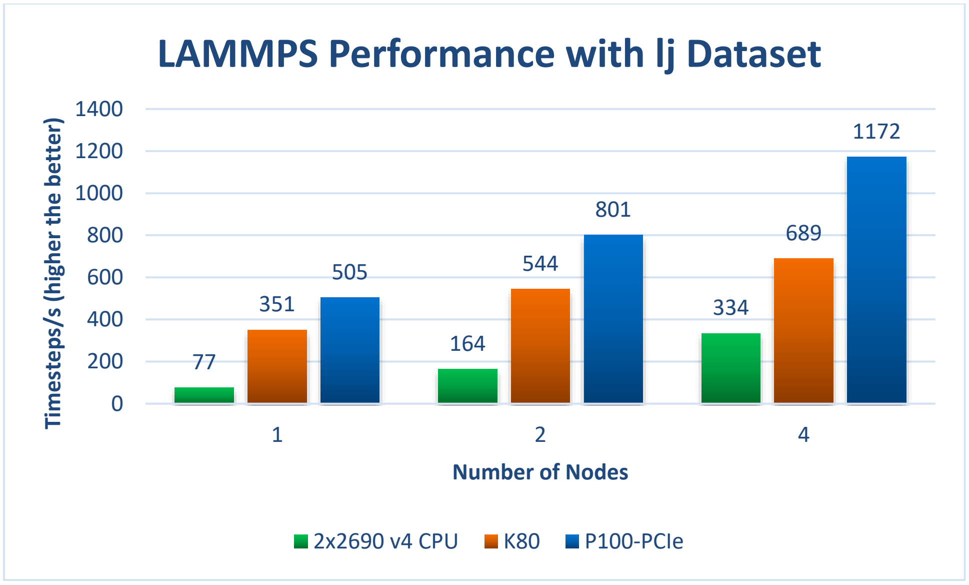

Figure 7: LAMMPS Performance Comparison

HOOMD-blue

HOOMD-blue (for Highly Optimized Object-oriented Many-particle Dynamics - blue) is a general purpose molecular dynamic simulator. Figure 8 shows the HOOMD-blue performance. Note that the y-axis is in logarithmic scale. It is observed that 1 P100 is 13.4x faster than dual CPU. The speedup of using 2 P100 is 1.5x compared to using only 1 P100. This is a reasonable speedup. However, with 4 P100 to 16 P100, the speedup is from 2.1x to 3.9x which is not high. The reason is that similar to LAMMPS, this application also involves lots of communications among all used GPUs. Based on the analysis in LAMMPS, using configuration B should reduce this communication bottleneck significantly. To verify this, we ran the same application again on a configuration B server. With 4 P100, the performance metric "hours for 10e6 steps" was reduced to 10.2 compared to 11.73 in configuration G, resulting in 13% performance improvement and the speedup compared to 1 P100 was improved to 2.4x from 2.1x.

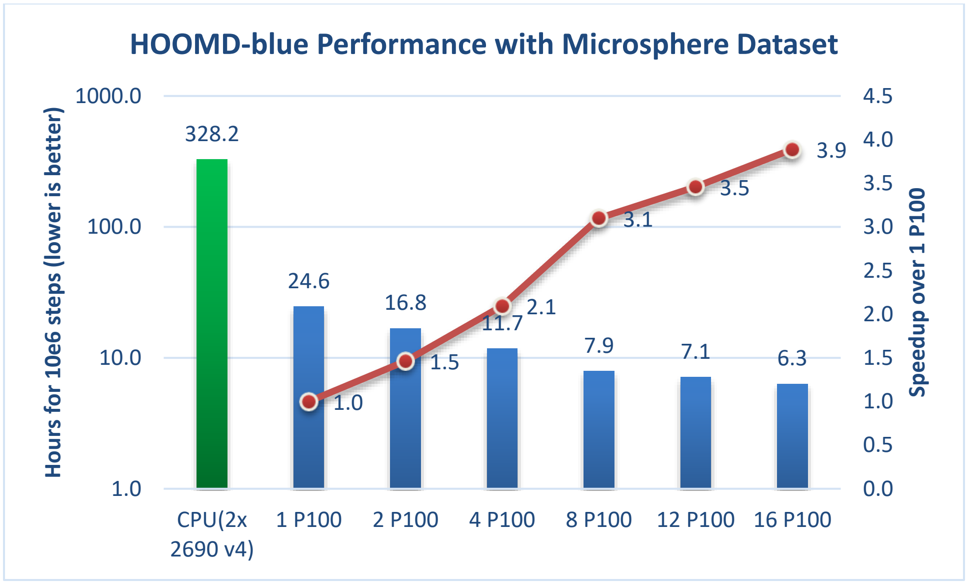

Figure 8: HOOMD-blue Performance on CPU and P100-PCIe

Amber

Amber is the collective name for a suite of programs that allow users to carry out molecular dynamics simulations, particularly on biomolecules. The term Amber is also used to refer to the empirical force fields that are implemented in this suite. Figure 9 shows the performance of Amber on CPU and P100-PCIe. It can be seen that 1 P100 is 6.3x faster than dual CPU. Using 2 P100 GPUs is 1.2x faster than using 1 P100. However, the performance drops significantly when 4 or more GPUs are used. The reason is that similar to LAMMPS and HOOMD-blue, this application heavily relies on P2P access but configuration G only supports that between 2 pair GPUs. We verified this by again testing this application on a configuration B node. As a result, the performance of using 4 P100 was improved to 791 ns/day compared to 315 ns/day in configuration G, resulting in 151% performance improvement and the speedup of 2.5x. But even in configuration B, the multi-GPU scaling is still not good enough. This is because when the Amber multi-GPU support was originally designed the PCI-E bus speed was gen 2 x 16 and the GPUs were C1060 or C2050s. However, the current Pascal generation GPUs are > 16x faster than the C1060s while the PCI-E bus speed has only increased by 2x (PCI Gen2 x 16 to PCI Gen3 x 16) and Infiniband interconnects by about the same amount. Amber website explicitly states that “**It should be noted that while the legacy MPI and GPU-Direct methods of multi-GPU communication are still supported, and will be used by the code automatically if peer to peer communication is not available, you are very unlikely to see any speedup by using multiple GPUs for a single job if the GPUs are newer than C2050s. Multi-node runs are almost impossible to get to scale.**” This is consistent with our results on multi-node. Because it is obvious to see that in Figure 9, the more nodes are used, the worse the performance is.

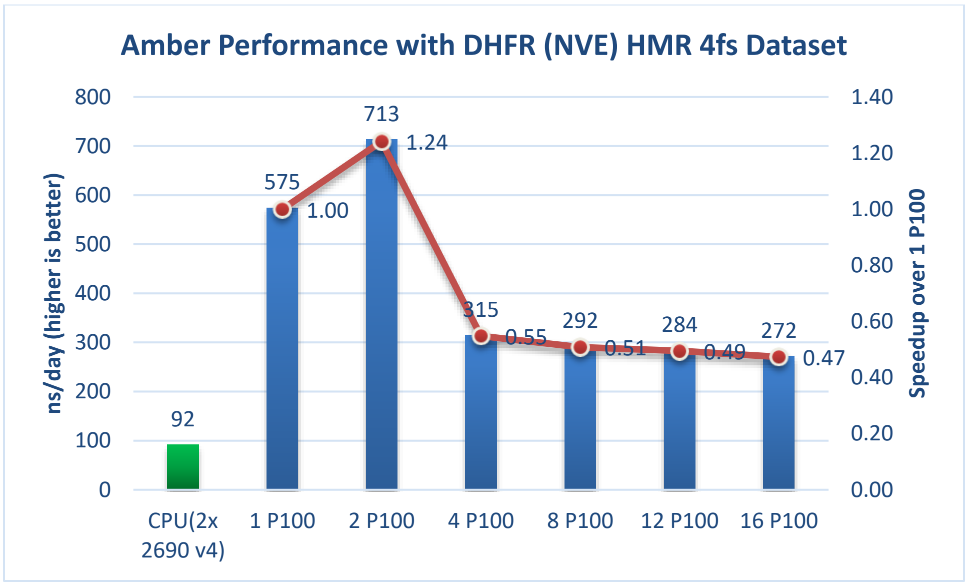

Figure 9: Amber Performance on CPU and P100-PCIe

ANSYS Mechanical

ANSYS® Mechanical™ software is a comprehensive finite element analysis (FEA) tool for structural analysis, including linear, nonlinear dynamic, hydrodynamic and explicit studies. It provides a complete set of element behavior, material models and equation solvers for a wide range of mechanical design problems. The finite element method is used to solve the partial differential equations which is a compute and memory intensive task. Our testing focused on the Power Supply Module (V17cg-1) benchmark. This is a medium sized job for iterative solvers and a good test for memory bandwidth. Figure 10 shows the performance of ANSYS Mechanical on CPU and P100-PCIe. It is shown that within a node, 4 P100 is 3.8x faster than dual CPUs. And with 4 nodes, 16 P100 is 2.3x faster than 8 CPUs. The figure also shows that the performance scales well with more nodes. The speedup with 4 nodes is 2.8x compared to 1 node.

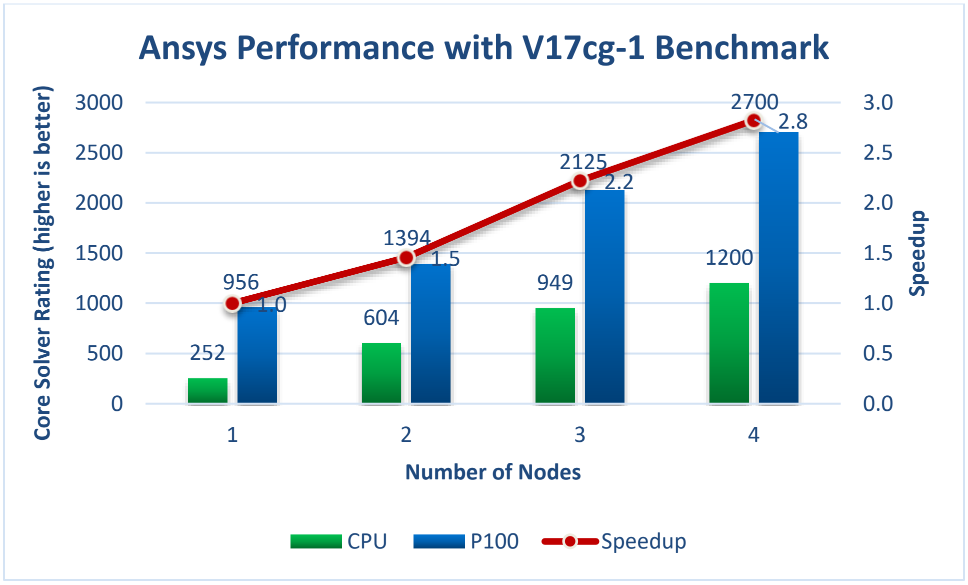

Figure 10: ANSYS Mechanical Performance on CPU and P100-PCIe

RELION

RELION (for REgularised Likelihood OptimisationN) is a program that employs an empirical Bayesian approach to refinement of (multiple) 3D reconstructions or 2D class averages in electron cryo-microscopy (cryo-EM). Figure 11 shows the performance of RELION on CPU and P100-PCIe. Note that y-axis is in logarithmic scale. It demonstrates that 1 P100 is 8.8x faster than dual CPU. From the figure we also notice that it does not scale well starting from 4 P100 GPUs. Because of the long execution time, we did not perform the profiling for this application. But it is possible that the reason of the weak scaling is similar to LAMMPS, HOOMD-blue and Amber.

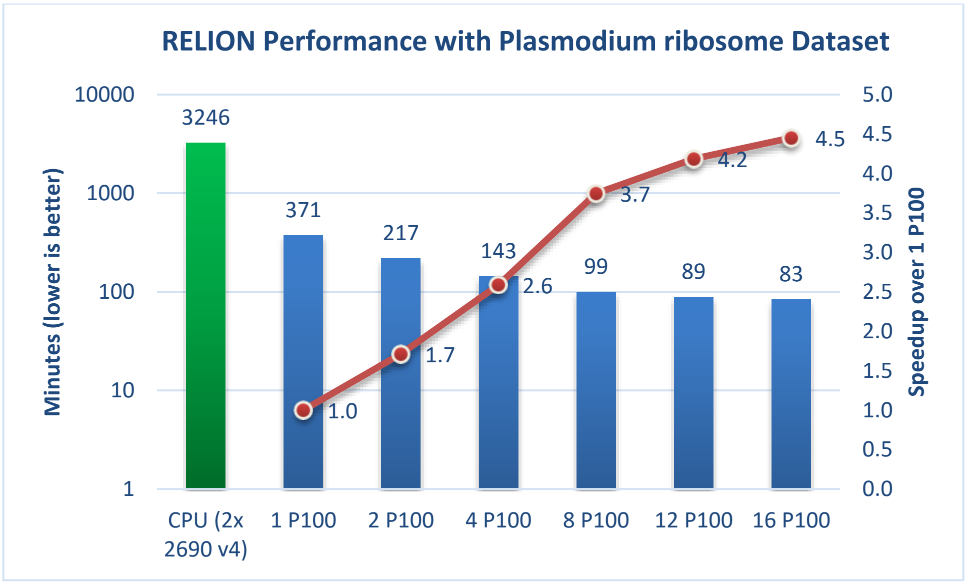

Figure 11: RELION Performance on CPU and P100-PCIe

Conclusions and Future Work

In this blog, we presented and analyzed the performance of different applications on Dell PowerEdge C4130 servers with P100-PCIe GPUs. In all of the tested applications, HPL, GROMACS and ANSYS Mechanical benefit from the balanced CPU-GPU configuration in configuration G, because they do not require P2P access among GPUs. However, LAMMPS, HOOMD-blue, Amber (and possibly RELION) rely on P2P accesses. Therefore, with configuration G, they scale well up to 2 P100 GPUs, then scale weakly with 4 or more P100 GPUs. But with Configuration B, they scale better than G with 4 GPUs, so configuration B is more suitable and recommended for applications implemented with P2P accesses.

In the future work, we will run these applications on P100-SXM2 and compare the performance difference between P100-PCIe and P100-SXM2.