Fermilab Multicore and GPU-Accelerated Clusters for Lattice QCD

Don Holmgren, Nirmal Seenu, James Simone, Amitoj Singh

Fermi National Accelerator Laboratory†
Batavia, Illinois, USA

1. Introduction

Lattice QCD (LQCD) uses large scale numerical simulations to study the strong interactions between quarks mediated by gluons (quantum chromodynamics, or QCD). In the United States, most lattice QCD theorists are members of the USQCD collaboration. Members of USQCD study Standard Model QCD problems in high energy and nuclear physics, as well as various theories beyond the standard model.

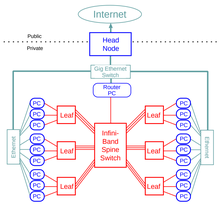
The Office of Science of the U.S. Department of Energy, through the high energy and nuclear physics program offices, has strongly supported LQCD through the SC LQCD (FY06-FY09), SC LQCD-ext (FY10-FY14) and LQCD ARRA (FY09-FY13) hardware projects. These projects have funded the purchase and operations of clusters dedicated to LQCD computations. Time on these resources is allocated by USQCD. Together with the advanced scientific computing research office, the high energy and nuclear physics program offices have also supported LQCD software development through the SciDAC (Scientific Discovery through Advanced Computing) and SciDAC-2 programs.

Fermilab currently operates four LQCD clusters. Three of these are conventional Infiniband clusters based on multicore processors, and the fourth is a GPU-accelerated cluster. Both multicore and GPU architectures have improved the price/performance of LQCD clusters.

2. Computational Requirements

Inversion of the Dirac operator (*Dslash*) is the most computationally intensive task of LQCD codes. Using the improved staggered action (*asqtad*) as an example, during each iteration of the *asqtad Dslash* inverter, eight sets of SU(3) matrix-vector multiplies occur using nearest and next-next-nearest neighbor spinors. The 3x3 matrices describe gluons, and the 3x1 spinors describe quarks. For a calculation using multiple nodes, ideally these floating point operations overlap with the communications of the hyper-surfaces of the sub-lattices held on neighboring nodes. The low ratio of floating point operations to bytes of memory read or written during *Dslash* execution results in a strong demand for memory bandwidth. Effective parallel machine designs for LQCD calculations need to balance floating point performance, memory and network bandwidths, and network latency. On any cluster, one of these will be the limiting factor determining performance for a given problem size.

Inversion of the *Dslash* operator is computationally equivalent to inverting a very large, very sparse matrix. For example, a $48^3 \times 144$ problem, typical of the size of current simulations, results in a complex matrix with 47.8 million $\times$ 47.8 million elements, of which 1.15 billion are non-zero (about one in every 2 million). Iterative techniques such as "conjugate gradient" are used to perform these inversions. Specific LQCD simulations, such as the computation of a decay constant, require many TFlop/sec-years of calculations using large-scale parallel machines. LQCD codes employ MPI or other message passing libraries for communications. Networks such as Infiniband provide the required high bandwidth and low latency. The figure below shows a typical LQCD cluster. A cascaded Infiniband network is used, with a large "spine" switch connected to "leaf" switches. All of the Fermilab LQCD clusters described here implement this design. On accelerated clusters, each of the nodes contains one to four GPUs.



3. LQCD Clusters at Fermilab

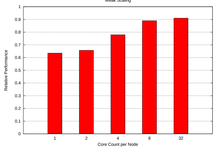
Name	CPU	Nodes	Cores	Performance
Kaon	Dual 2.0 GHz Dual-Core Opteron 240	300	1200	4.3 GF/node
JPsi	Dual 2.1 GHz Quad-Core Opteron 2352	856	6848	9.8 GF/node
Ds	Quad 2.0 GHz 8-Core Opteron 6128	421	13472	50.9 GF/node
Dsg	Dual 2.53 GHz Quad-Core Intel E5630	76	608	23.1 GF/node
	Dual NVIDIA M2050	*152*	*68096*	*93.4 GF/GPU*

The table above lists the currently operational Fermilab LQCD clusters. The performance values listed are sustained GFlops on LQCD parallel applications using 128 cores. GPU performance varies widely across LQCD applications; the value shown for the Dsg cluster corresponds to *isotropic clover* code.

4. Multicore Processors and LQCD

LQCD codes scale very well on multicore processor clusters:

- All codes have platform-specific message-passing implementations, using MPI on clusters and native communications on other systems (*e.g.* BlueGene).
- In MPI applications, LQCD generally uses one rank per core. Early threaded implementations did not result in significant performance improvements over this naïve use of MPI.
- MPI launchers must be NUMA (Non-Uniform Memory Access) aware, as the principal bottleneck is memory bandwidth.
- Weak scaling (relative performance as node counts are increased proportional to problem size) has steadily improved with higher core counts.

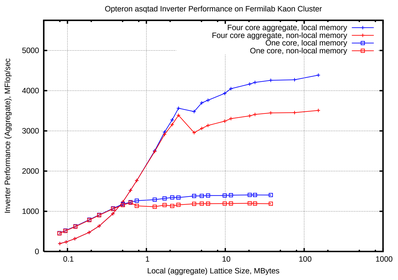


5. Benchmarking

Fermilab awards purchase contracts to computer vendors according to a best value process. Price/performance, in dollars per sustained MFlop, is a key component in determining the value of a vendor's proposed system. For each of the JPsi, Ds, and Dsg purchases, vendors were provided with benchmark packages that reported single-node performance of LQCD applications and micro-benchmarks. The benchmark package for the Dsg cluster included both GPU-accelerated and non-accelerated applications. The benchmarks can be run on systems with a wide range of processor sockets per node and cores per socket; the included MPI launcher automatically tailors execution to match the NUMA design.

6. NUMA Effects

All currently available multi-socket multicore commodity computers have NUMA architectures. Since memory bandwidth limits LQCD application performance, binaries should either be built using NUMA-aware MPI libraries, or invoked with NUMA controls. The figure below shows the performance degradation that occurs on LQCD codes that use non-local memory. Fermilab provides all users with *numactl*-based shims that force non-NUMA-aware MPI launchers to bind MPI ranks to cores and to set strict local memory allocation policies.



7. GPU-Accelerated LQCD

- Lattice theorists in Europe started using GPUs in 2005 (OpenGL with CG!).
- Barros *et al.* at Boston U. implemented a Wilson-Dirac operator in CUDA in 2007 with about a 10-fold speedup over conventional x86 processors.
- This Boston U. work evolved under the SciDAC-2 program into *QUDA*.
  - Initially a single GPU code for Wilson-Dirac, *QUDA* now supports parallel execution across multiple GPUs and multiple hosts, as well as including implementations for the clover, twisted-mass, staggered, and HISQ actions.
  - Mixed-precision algorithms are used to improve the performance of some actions.
  - GPUs are relatively more memory-bound for LQCD codes than CPUs. For some actions, utilized memory bandwidth is minimized through reduced representations (*i.e.* the exploitation of SU(3) symmetries).

8. The Dsg GPU-Accelerated Cluster

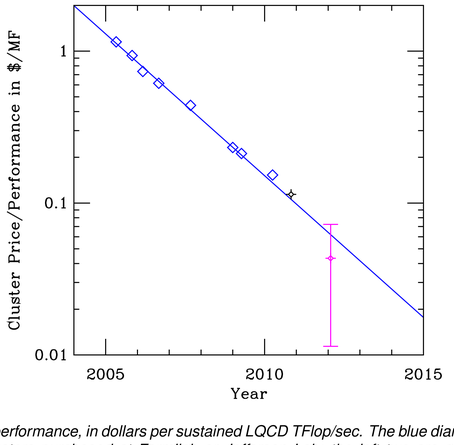
In 2011, Fermilab designed and purchased a GPU-accelerated cluster with optimal strong-scaling performance on calculations that require many GPUs operating in parallel. To meet this requirement, vendor-proposed host systems were restricted to those that provided sufficient PCI Express bandwidth to support the installed GPUs (16 gen2 lanes per GPU) and quad-data-rate Infiniband (8 gen2 lanes per HCA). Hardware details of this cluster:

- Vendor: Hewlett-Packard, using SL390s G7 blade server hosts
  - Dual socket, 4 cores/socket, 2.54 GHz, Intel "Westmere" processors
  - 48 GBytes memory and 2 GPUs per host, 76 hosts, 152 GPUs in total
- GPUs:
  - NVIDIA Tesla M2050, 1 TFlop/sec peak single precision, 448 cores
  - 3 GBytes memory per GPU, ECC-capable, hardware double precision

Large-scale ($96^3 \times 192$ and $64^3 \times 96$) gauge configuration generation has been demonstrated on this cluster using parallel runs with 128 GPUs. The performance of GPUs varies with the specific LQCD application. In production running, users have reported application-dependent speed-ups of between 2.1 and 13.3, comparing Ds node hours to Dsg node hours for performing equivalent calculations.

9. Decreases in Price/Performance for LQCD Clusters

Fermilab has tracked LQCD price/performance data since 1995. The figure below shows the price/performance observed on production Infiniband clusters deployed since 2005. The fit uses data points between 2005 and 2010 and has a halving time of 1.613 years. Between 1995 and 2004, clusters based on Myrinet switched networks, and on gigabit Ethernet toroidal mesh networks, exhibited a faster halving time of 1.2 years. Higher core counts per node have compensated for the cessation of increasing processor clock speeds that occurred in the middle of the last decade. Based on data from Dsg, the use of GPUs results in price/performance that significantly surpasses the long term trend.



Price/performance, in dollars per sustained LQCD TFlop/sec. The blue diamonds are clusters purchased at Fermilab or Jefferson Lab; the left two are clusters based on single core computers, and the others are clusters based, respectively, on 2, 4, 8, 8, 8, and 8 cores per node. The black star is the Ds cluster with 32 cores per node. The pink range shows the cluster equivalent price/performance for codes running on Dsg that achieve speed-ups of between 2.1 and 13.3.

†Operated by Fermi Research Alliance, LLC under Contract No. De-AC02-07CH11359 with the United States Department of Energy.