ARRA LQCD Computing

Technical Design & Performance

Chip Watson

May 10, 2011

LQCD ARRA Technical Goals

Performance Goal:

To significantly increase the computing resources available to the USQCD collaboration for "analysis"…

Original target was 16 Tflops sustained aggregate performance averaged over the 3 dominant inverter actions:

- Domain Wall Fermions (DWF)
- Staggered (asqtad = a-squared tadpole)
- Clover, particularly anisotropic clover

As a slight variation from the LQCD-ext project, all three actions are included in the benchmark definition.

Quantifying Aggregate Performance

(Reminder) LQCD computing proceeds in 2 phases:

1. Configuration generation (on supercomputers)
   - ❖ Must be produced sequentially, at highest performance
   - ❖ End product: 1000+ configuration files
2. Analysis (propagator generation + observables)
   - ❖ 1000 + jobs able to run in parallel
   - ❖ Target performance: 1% of configuration generation (then at 10's of Tflops)

Analysis is the task relevant for this project. For benchmarking for the LQCD ARRA resources, we selected production lattice sizes for each of the 3 inverters:

- ❖ Anisotropic Clover: 24^3 x 128
- ❖ Asqtad: 56^3 x 96
- ❖ DWF: 32^3 x 64 x 16

Inverters were run on a single node with a fraction of this volume in order to project the performance of a multi-node job achieving > 0.25 Tflops

GPU vs. CPU

Strategy: buy as much computing capacity for the dollar as possible.

Challenge: setting the split between GPU (highest performance) and CPU (greatest flexibility)

Phase 1: 25% of compute funds to GPU

- ✓ Enough software was becoming ready to exploit this capacity, and software development environment (CUDA) was maturing rapidly
- ✓ Impact on non-GPU capacity for USQCD was minimal (15%)
- ✓ GPUs allowed this project to double the USQCD total computing capacity

Phase 2: 50% of compute funds to GPU

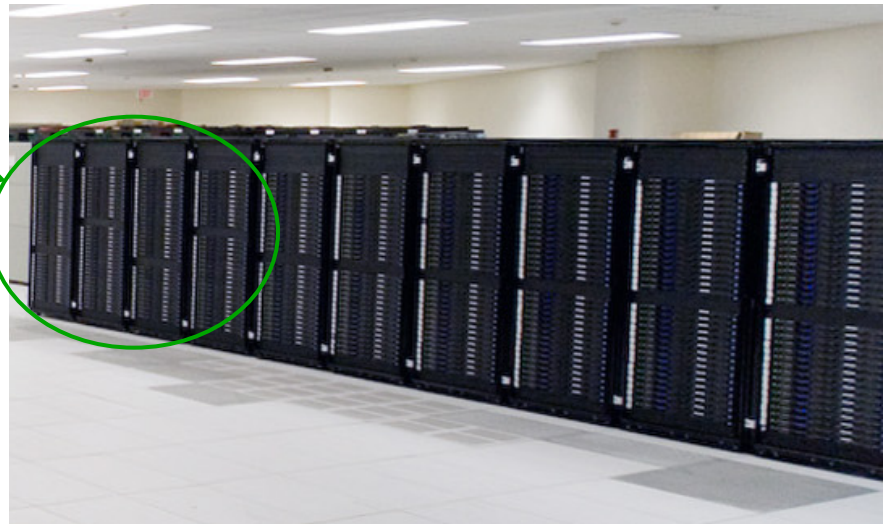
- ✓ Multiple groups were in production, and were eager to absorb a large increase in capacity (allowed project to again double the USQCD total capacity)
- ✓ Availability of ECC memory on the GPUs held a promise of expanding beyond inverters to satisfy more of the collaborations computing requirements
- ✓ Choice of 25% vs. 50% was only a 10% effect on non-GPU, and once the first phase of the LQCD-ext IB cluster at Fermilab came online only a few months later, this impact was further reduced to about 7%

CPU Cluster & IB Fabric Design

The most cost effective conventional nodes were dual Intel systems,
2.4 GHz Nehalem / 2.53 GHz Westmere (phase 1 / 2),
giving about 20 Gflops/node, so needed 16 nodes for a job.

QDR Infiniband switches have 36 ports, so can hold 32 nodes and still have ports free to connect to the file systems (powers of 2 are best for LQCD). Deploying multiple sets of 32 nodes reduces the cost of the Infiniband fabric while maintaining the highest efficiency for jobs up to 640 Gflops.

17 racks purchased for phases 1 & 2:
13 as single racks non-oversubscribed,
4 interconnected 2:1 oversubscribed
(to support job up to ~2 Tflops)



Note: homogeneous fabric with 2:1 oversubscription would have required 13 additional switches and 200 cables, and would have had somewhat worse multi-node scaling, yielding 5%-10% lower performance per dollar. A few extra nodes on the fabric solved the problems of 1 or 2 failed nodes preventing large jobs from running.

All racks have 2 uplinks to a core switch for file services

File Servers

Planned capacity, performance & budget:

280 TB at $1K / TB, > 1 GB/s, $280K

Conservative cost estimate: higher than Fermilab had been getting in large bulk procurements associated with the Tier 1 center but lower than Jlab's recent buys.

Final Configuration: 416 TB, > 2 GB/s, $228K

Phase 1: 224 TB across 14 servers

- dual Nehalem 2.26 GHz, 12 GB memory
- 24*1TB disks, 24 disk RAID controller, DDR Infiniband
- bandwidth measured at 1.4 GB/s using 6 nodes (single DDR uplink)

Phase 2: 192 TB across 4 servers

- similar to above, but with 3 RAID-6 (8+2) strips per server instead of 2
- 2 TB disks, QDR Infiniband, higher performance RAID controller
- somewhat lower bandwidth / TB, but still more than necessary

GPU Cluster Design

Design Parameter Space:

- NVIDIA or ATI

  LQCD software use NVIDIA's CUDA language, so NVIDIA selected

- NVIDIA options:
  - Phase 1: GT200b chip: Tesla card or 4 GPU array, or gaming card

    Tesla line offered larger memory and promise of higher quality at very high price; quality issue wasn't so relevant for inverters so gaming cards selected

  - Phase 2: Fermi chip: Tesla card, or gaming card

    Tesla line now offered ECC memory, useful for non-inverter applications; a mix of card types was chosen

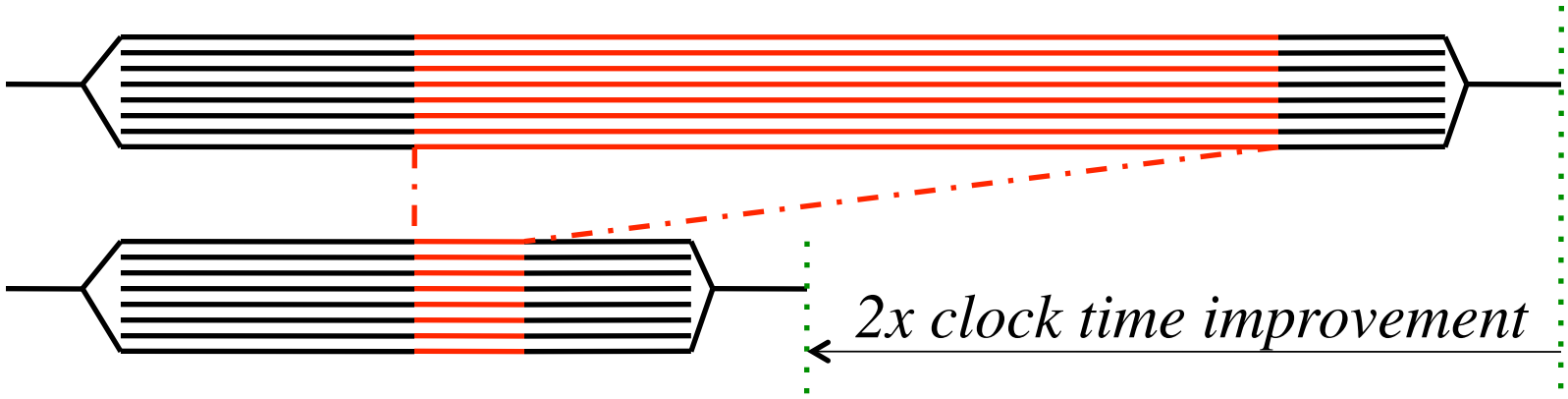
- Number of GPUs per host (i.e. per CPU)
- GPU I/O bandwidth: single or dual Intel chipsets on host
- Networking (multi-node jobs or just single node)

Amdahl's Law (Problem)

A major challenge in exploiting GPUs is Amdahl's Law:

If 60% of the code is GPU accelerated by 6x,

the net gain is only 2x.



Also disappointing in this scenario: the GPU is idle 80% of the time!

Fortunately many LQCD codes spend > 95% of their clock time in a single kernel, a matrix inversion, and so for these applications Amdahl's Law was not (yet) a show-stopper.

Ultimate solution: we need to move more code to the GPU, and/or need task level parallelism (overlap CPU and GPU).

Design Tradeoffs

Cost optimization:

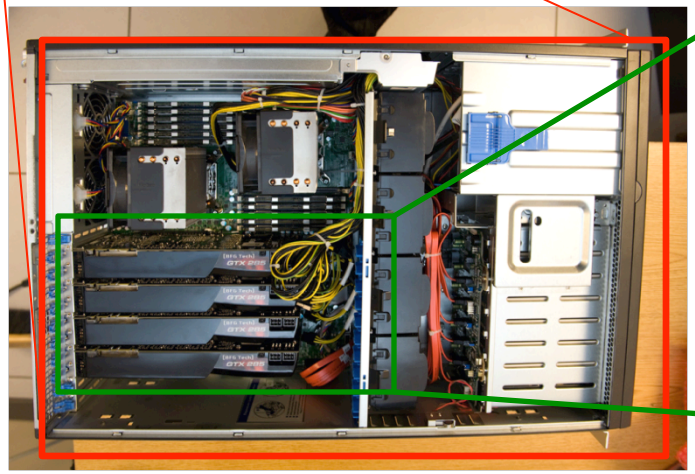
- Host is ~$4K, so 4 GPUs per host better amortizes that cost than just 1 or 2, but worsens the effect of Amdahl's Law (acceleration is higher)
- For jobs primarily running inversions, 4 GPUs would be more cost effective than just 2. Prior to phase 1 award we identified 2 classes of applications that were suitably inverter heavy (thus capable of exploiting 4 per node). Additional applications became ready in time for phase 2, allowing for an even greater deployment of 4 GPU nodes
- I/O bandwidth: suitable dual chipset motherboards appeared just in time for us to consider them for 4 GPU

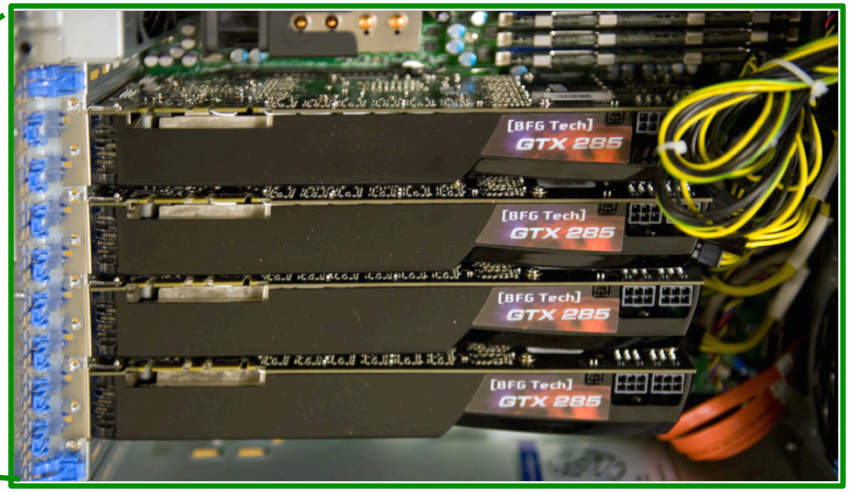
Conclusions:

- Buy mostly quad GPU nodes (cost effective)
- Buy some dual GPU nodes with Infiniband for R&D for future larger problems and/or more CPU per GPU

Serendipity: after phase 1 award we were able to switch from 1 GB gaming cards to 2 GB gaming cards (we had a helpful vendor). This product soon disappeared from the marketplace.

Phase 1 GPU Hardware





Host:
- 2.4 GHz Nehalem
- 24 or 48 GB memory / node (mixed)
- 65 nodes, 200 GPUs

Original configuration:
- 40 nodes w/ 4 GTX-285 GPUs (2 GB)
- 16 nodes w/ 2 GTX-285 + QDR IB
- 2 nodes w/ 4 Tesla C1050 or S1070
- 7 nodes with no GPU for later R&D

Phase 1 Hardware Summary

Infiniband Cluster:

320 nodes, 5.5 Tflops aggregate on LQCD codes

dual quad core Nehalem 2.4 GHz, 24 GB memory

QDR (quad data rate, 40 Gbps) Infiniband

GPU Cluster:

65 nodes, 200 GPUs

16 dual GPU, QDR Infiniband

42 quad GPU

7 empty (future R&D, alternate GPU cards)

192 of the 200 GPUs were gaming cards, GTX-285

Host nodes all had the same specs as the conventional cluster (above) except some had 48 GB memory & all had dual 5520 chipsets to support 4 GPUs or Infiniband cards

File Servers:

224 TB Lustre system, on Infiniband, >1GB/s

Phase 2 Hardware Summary

Infiniband Cluster:

224 nodes, 4.0 Tflops aggregate

- dual quad core Westmere 2.53 GHz
- 24 GB memory
- QDR Infiniband
- all racks configured as 32 nodes, no oversubscription
- all nodes capable of holding one GPU (future upgrade)

GPU Cluster:

52 nodes, 338 GPUs (all NVIDIA Fermi)

- 16 quad GPU, QDR Infiniband in half bandwidth slot
- 36 quad GPU
- 32 additional GPUs to upgrade the Phase 1 duals to quads
- 28 additional GPUs to fill the Phase 1 empty nodes
- 36 additional GPUs to go into one rack of the Phase 2 Infiniband cluster (above)

128 of the GPUs were Tesla Fermi (with 3 GB ECC memories) => 32 quad servers w/ half QDR

GPU hosts have same specs as the IB cluster above except that they all have 48 GB memory, and only some have QDR IB. Also: all Phase 1 GPU nodes were upgraded to 48 GB memory.

File Servers:

Additional 192 TB added to Lustre

GPU Comparison

Card	GPU	#cores	clock speed (GHz)	memory size (GB)	raw memory bandwidth (GB/s)	clover inverter (Gflops)[1]	cost
GTX-285	GT200b	240	1.47	2	159	135	$500
C1060	GT200b	240	1.30	4	102	100	$1500
GTX-480	Fermi	480	1.40	1.25	177	270	$500
C2050[2]	Fermi	448	1.15	2.67	144	185	$2100

[1] Newest development code gets up to 310 Gflops on GTX-480; data is this talk uses older 270 Gflops; all numbers are for mixed precision

[2] C2050 evaluated with ECC enabled

The Fermi Tesla line of cards (C2050) has a significant advantage in having ECC memory so that more than just inverters can be safely executed. This comes at a steep price: 4x on GPU price, and 1.5x on lower performance. Integrated into a host this yields a price performance difference between the two of 3x.

Conclusion: judicious use of gaming cards is a very good idea as long as we have inverter heavy loads (which we do).

GTX-480 Problems

The 192 GTX-285 cards we bought in phase 1 were very stable, and reported no errors in running a memory test.

The 210 GTX-480 cards did much worse:

86	encountered no errors in a 2 hour test
80	encountered 1-10 errors in 2 hours
26	encountered memory errors at 1-10 per minute
2	encountered memory errors at about 1 per second
4	undetected by the NVIDIA driver
2	bad fan & hung running CUDA code
10	hung running CUDA code

We were an early buyer of GTX-480 cards for computing, and apparently caught some early quality control issues.

The first two sets, 166 GPUs, were put into production use.

GTX-480 Problem Resolution

1. The manufacturer PNY wasn't helpful, even with cards that would not run at all. Only 2 were replaced under manufacturer warranty.
2. Jlab tried several other memory test programs, tried under-clocking the poor cards, all to no avail. LQCD software Chroma/QUDA, however, ran successfully on all functioning cards, despite the memory errors (evidence that low error rates are not a problem).
3. We developed a more rigorous testing procedure, running a 2 hour test on every GPU every week to catch any further degradation, and removing from the production queue any GPU with more than 10 errors in 2 hours. Users were warned that the cards were only suitable for inverters, and that applications should test inversion residuals.
4. The vendor Koi eventually agreed to replace 35 cards with new cards from ASUS. All but 1 of these passed our tests with low error rates.
5. Today 4*45 = 180 GTX-480s are in production. Memory testing is ongoing, and has caught one GTX-285 and one C2050 failure.

Gaming GPUs: An Early Taste of Exascale

Reliability *– System architecture will be complicated by the increasingly probabilistic nature of transistor behavior due to reduced operating voltages, gate oxides, and channel widths/lengths resulting in very small noise margins. Given that state-of-the-art chips contain billions of transistors and the multiplicative nature of reliability laws, building resilient computing systems out of such unreliable components will become an increasing challenge. This cannot be cost- effectively addressed with pairing or TMR; rather, it must be addressed by X-stack software and perhaps even scientific applications.*

-- from The International Exascale Software Project Roadmap
http://www.exascale.org/

Key GPU Decision Points (past and future)

1. Balance between Tesla (ECC) and GTX
   - GTX is 3x more cost effective for single precision inverters on single box
   - Single precision was to be a large fraction of running for the coming year

2. Balance between single node, and multi-node running (i.e. Infiniband)
   - Multi-node is needed for performance greater than 1 Tflops in the inverter
   - Multi-node is also needed for larger problems, over 10 GB in the GPUs

     32^3x256 just fits into 4 Fermi GPUs, but needs 96 GB host memory, which is only affordable as two nodes of 48 GB, hence a need for IB

     But: adding a QDR HCA precluded running 4 GPUs in a (commodity) box, dropping to 2 GPUs increased cost per flop by 33% (more "box+cpu" cost overhead, amortized over fewer GPUs)

3. Balance between GPU and CPU

   Codes with only a portion of the code ported to the GPU can profit from having only a single GPU (i.e. more CPU per GPU). This is not as big a win per GPU, but still more optimal than no GPU.

Blurring the Boundary

The Phase 2 Infiniband cluster was configured so that each node can hold 1 GPU, giving a very cost effective way to add many GPUs, but this "consumes" standard cluster nodes (but increases their performance considerably).

If $500 GTX cards are placed into these nodes, then the cost of NOT using the GPU is rather small, about 12% of the total node cost. But when the GPU load is very heavy, the capability is there to use, and could yield a 2x-6x performance boost (for 1 GPU).

Current Status: of the 17 racks of the conventional cluster nodes (Phase 1 + 2), 1 rack (32 nodes + 2 spares) have been upgraded to include a GPU. This set of 32 nodes turned out to be valuable for one project that needed >50 GB of GPU memory to hold their problem.

GPU Job Effective Performance

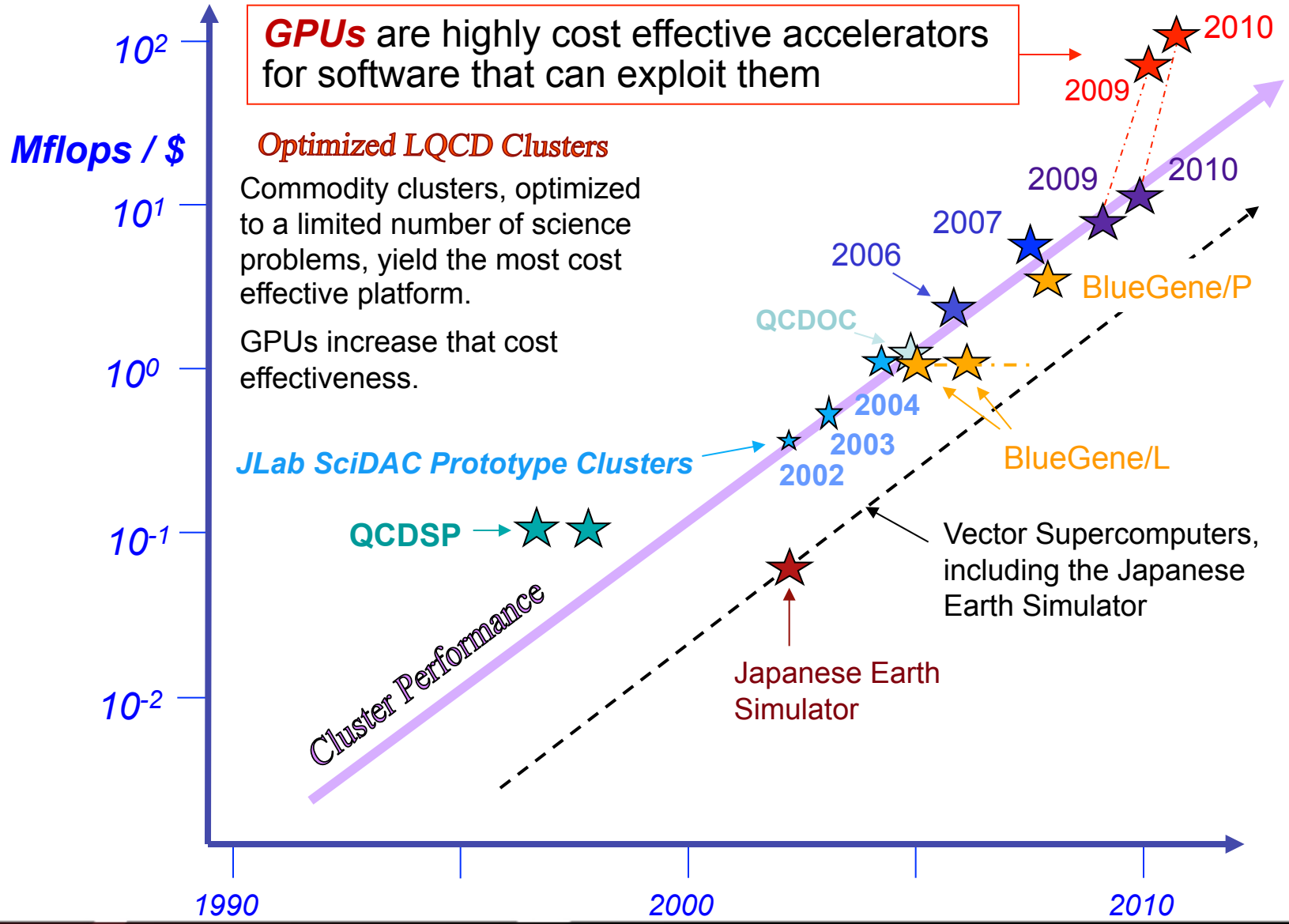
Comparing GPUs to regular clusters can't be done on the basis of inverter performance (Amdahl's Law problem), so instead we compare job clock times, and from that derive an "effective" performance, which is the cluster inverter performance multiplied by the job clock time reduction.

The following table shows the number of core-hours in a job needed to match one GPU-hour in a job. Last project used 32 single GPU nodes and was I/O bound.

The allocation-weighted performance of the cluster is **63 TFlops**.

Project	2010-2011 Hours	#GPUs, nodes	Jpsi core hours / GPU hour (job time)	Effective Performance Gflops/node	GPU used
Spectrum	1,359,000	4, 1	180	800	(average)
thermo	503,000	4, 1	90	400	(average)
disco	459,000	4, 1	92	410	C2050
Tcolor	404,000	4, 1	40	175	GTX285
emc	311,000	4, 1	80	350	(average)
gwu	136,000	32, 32	47	50	GTX285

Science per Dollar for LQCD Applications



Technical Summary

The ARRA LQCD Computing project has deployed

10 Tflops conventional infiniband systems

416 TBytes disk, backed by multi-petabyte tape library

508 GPUs equivalent to 100 Tflops sustained capacity for anisotropic clover inverter-heavy jobs, and 63 Tflops for the mix of jobs running this year

Total deployed capacity: 73 Tflops (effective), a gain of 4.5x over the original plan of 16 Tflops.

The total effective Tflops depends upon the efficiency with which the applications use the GPU, and could rise as a larger fraction of the existing code is ported to the GPU (reduced Amdahl's Law problem), or fall as new applications with lower GPU intensity begin to exploit the GPUs.