LIGO Computing Optimization

Peter Couvares, LIGO Laboratory/Caltech

Condor Week 2015
May 21, 2015

Who am I? What is LIGO?

- Former Condor Team member (’99-’08).
- Until recently at Syracuse University (’10-’15) focused on distributed computing problems for the LIGO Scientific Collaboration, and fostering a research computing community at SU.
- Now working directly for the LIGO Laboratory/Caltech, with a focus on data analysis computing, particularly optimization.
- LIGO (the Laser Interferometer Gravitational-Wave Observatory) is a large scientific experiment to detect cosmic gravitational waves and harness them for scientific research.
- http://ligo.org/

Two Kinds of Optimization

- HPC — how to achieve more FLOPS on a core or socket
- HTC — how to achieve more *goodput-contributing* FLOPY on a distributed network of heterogenous clusters managed and used by heterogenous humans
- **FLOPY != FLOPS * 365 * 24 * 60 * 60**

HPC vs HTC

FLOPY =
(HPC factors: FLOPS * 365 * 24 * 60 * 60) /
(HTC factors for: un-utilized computing resources, job and workflow scheduling overhead, job and workflow scheduling unreliability, human mistakes, re-runs, time spent babysitting, priority inversions, etc.)

- I've spent a career focused on the denominator.
- But… it turns out FLOPS is still a first-order term in the numerator! Who knew?!

My Last Year

- LIGO got thwacked by the NSF for proposing to spend a lot of money on computers without showing that our data analysis software pipelines were well-optimized and well-tailored for the resources on which we planned to run them.

- Turns out they weren't—at least not by HPC standards at that scale.

- After a tremendous (ongoing) HPC-style optimization effort by many LIGO scientists and staff over the past year (with help from XSEDE staff @ TACC), we achieved a factor of ~8 reduction in estimated computational cost of our dominant data analysis pipeline, and a factor of ~6 in our next most expensive pipeline. More work is in progress, and additional gains are likely.

- **7/8 of a very big number… is a very big number.** The NSF is happy, LIGO is happy, kumbaya — as long as we don't take our foot off the gas.

LIGO Computing Optimization Project

- Dedicated and expert LIGO Lab and LSC staff devoted to optimization.
- The developers of every LIGO search pipeline engaged.
- Regular scientific and computing management body attention.
- Optimization team developed good working relationships with key scientists and developers.
- Given the optimizations achieved over the past year by our most expensive codes, we have a clear roadmap for further improvement of those and other computationally expensive pipelines.

HPC/FLOPS Optimization

- With effort and efficiency targets scaled to estimated computational cost, **each LIGO software pipeline** is being reviewed for potential for:
  - Scientific optimizations: reconsideration of search parameters in terms of computational cost / scientific benefit.
  - Algorithmic optimizations: FFT, BLAS, custom search code.
  - Compiler optimizations: gcc, icc, native instructions.
  - Vectorization, SIMD optimizations: both automatic and hand-tuned; looking ahead to AVX-512.
  - Multi-threading optimizations: single-core vs all-core testing to estimate potential benefits of cache monopolization.
  - GPU/MIC acceleration: high-level libraries and custom code.
- **Our work over the past year has shown how fruitful each of these can be.**

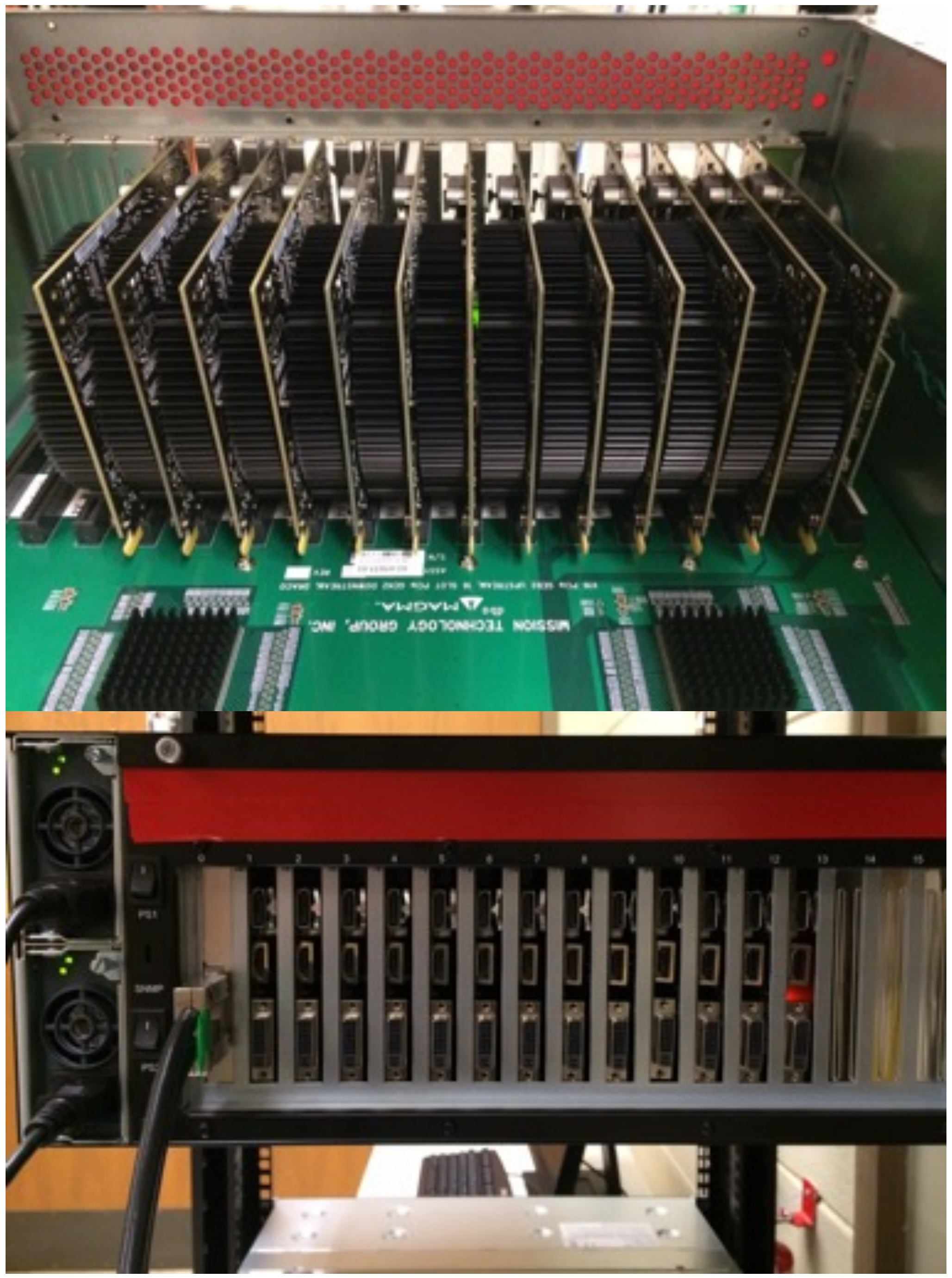
Hardware Test Stands (CPU)

- CPU Testbeds @ Caltech, Syracuse University, and AEI-Hannover
  - Diverse range of AMD, Intel Westmere, Sandybridge, Haswell and Atom-based CPUs.
  - Available for dedicated use by search pipeline developers for optimization, benchmarking, profiling.
  - Supports trade study of metrics relevant to search code performance and deployment scenarios:
    - wide range of clock speeds, CPU and core parallelism, cache sizes, instruction sets, power efficiency, physical density, cost, etc. enables better understanding of the effects of each on code performance and cost-efficiency.

Hardware Test Stands (GPU)

- GPU and MIC Testbeds @ Caltech and Syracuse University
- **Working with NVIDIA on CUDA driver enhancements to improve commodity GPU performance**:
  - e.g., mixed FP16/FP32 FFT: calculations in FP32, storage in FP16
  - LIGO being used as example use case for internal NVIDIA presentations.

GPU	Memory Bandwidth (GB/sec)	Price (USD$)/ unit	GB/s/ $1.00	Power/ unit	GB/s/ Watt
GTX 980	224	$555	0.40	165 W	1.36
GTX 970	224	$329	0.68	145 W	1.54
GTX 960	112	$200	0.56	120 W	0.93
GTX 760 No longer available	192	$300	0.64	170 W	1.13
GTX 750 Ti	86	$139	0.62	60 W	1.43
GT 730 DDR5	40	$75	0.53	25 W	1.60
GT 580 No longer available	192	$225	0.84	244 W	0.79
Tesla m2090 No longer available	177	$700	0.26	225 W	0.79
Tesla K80	480	$5,000	0.09	235 W	2.04
Tesla K10	320	$2,000	0.16	225 W	1.42



GPU/MIC Acceleration (cont.)

- Preliminary GPU results for our dominant-cost pipeline, soon to be reviewed:
  - **Best-performing GPU card (GTX 980) delivers *a factor of 3 higher* search throughput than the best-performing CPU socket (E5-1660 v3).**
  - Initial consumer GPU RAM reliability testing encouraging, but even with worst-case redundant execution, GPUs may hold a significant cost-efficiency advantage.
- Naive “offloading” of generic algorithms to co-processors via standard libraries may benefit other search codes; custom implementations hold more promise at additional development cost.
- Co-processor optimization work can lead to a better understanding of optimization landscape more generally, leading to improvements in CPU code performance. (E.g., consideration of in-place vs out-of-place FFTs.)

Optimization Approach: "The Whole Patient"

Iterate

- Performance Benchmarking and Computing Cost Estimation
- Optimization of Scientific Search Parameters
- Optimization of Data Analysis Methods and Algorithms
- Optimization of Implementation (via Library, Compiler, Hand)
- Hardware Trade Study, "Just in Time" Procurement
- LIGO-Virgo Computing Network Scheduling Optimizations
- Workflow Management & Robustness Optimizations
- Flexibility & Opportunism Optimizations (run anywhere!)
- Development, Testing, and Simulation Process Optimizations
- Optimization Sustainability (Pipeline Reviews, Regression Testing)
- Documentation, Training, Collaboration and External Engagement

Practical HTHPC Scheduling Issues

- How to schedule CPU sockets (rather than cores) and ensure processes/threads never cross sockets.
- How to schedule GPUs.
    - How to co-allocate CPU cores and GPUs, especially on high-density CPU systems where # GPUs ~= # of cores?
    - Virtualization?
- Soon: how to efficiently schedule workflows on a heterogenous mix of per-socket or per-GPU resources and single-core resources
    - Right now now a single workflow instance can't easily use both opportunistically — you need to choose in advance. With enough work in the queue maybe this isn't a problem—but it feels like something scientists should't have to think about.
    - How to keep single-core jobs from starving per-socket jobs.

Optimization Challenges

- When does an investment in FLOPS start to return less than one in FLOPY, e.g.:
    - scheduler robustness?
    - workflow portability or workflow simplicity? (these two can be in conflict!)
    - the capability to checkpoint & resume?
- How to measure computational efficiency:
    - actual vs peak FLOPS of critical code kernels
    - CPU utilization over process lifetime
    - CPU utilization over job lifetime (includes scheduling and i/o overhead on compute node)
    - goodput vs badput of a job
    - goodput vs badput of a DAG/workflow (i.e., DAG re-runs… does anyone measure this?)
    - CPU core hours per year per science task.
    - Science results for $$$ spent on labor and hardware over some time period.
    - ???

LIGO and XSEDE/OSG

- LIGO wants to remain engaged with XSEDE:
  - So we're prepared to scale quickly if we need to. (E.g.,"time to science".)
  - So we can continue to leverage XSEDE HPC expertise (ECSS).
  - So we're ready for (and can help define) evolving HTHPC computing models, post-O3.
- LIGO wants to re-engage with OSG:
  - To bridge the LIGO-Virgo Computing Network (LVCN) to campus grids that wish to contribute computing resources to LIGO.
  - To enable sharing of short-term LVCN surpluses.
  - To drive portability of our data analysis software pipelines to non-LVCN resources.
  - To simplify future LVCN resource configuration and management.

LIGO View of XSEDE HPC Resources (2014)

LIGO View of LIGO HTC Computing (2014)

LIGO Plans for HTHPC Computing (2015)