Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

Scientific Computing working group

Oli Gutsche and Gabe Perdue

2017 / May / 4

General points

- Underlying duality
  - Computing is essential to science - our science cannot be done without MASSIVE scale computing
    - In the future, data volumes will grow exponentially, making this even more concrete.
      - We will not be able to do physics without substantial advancements in computing.

- Investment in software, technologies, and computing hardware is crucial.

- To do our science, we will need to push the envelope in ways that require new hardware architectures and new technologies.
  - It is no longer feasible to ask for "do what I am doing now, but faster".

- New architectures and technologies are demanding new thinking and a revolution of the way we do analysis.

Thrusts

- We asked the Fermilab Community to help us formulate trends and directions and gather ideas. We tried to structure this into 3 thrusts

- Thrust 1: data access
  - Data and compute are both distributed and not necessarily co-located. How do we manage this?
  - How do we handle paradigms like those required by machine learning where training data access is very different from analysis (ntuple) data access?
- Thrust 2: scientific software
  - How do we adjust to a massively parallel world?
  - How do we handle new paradigms like machine learning?
  - We need to rethink how we do algorithms. Do we have to have deterministic algorithms? What matters in the end for us is that we understand our *systematic uncertainties*.
- Thrust 3: security and collaboration
  - We must work together with many people across administrative boundaries. How do we do that conveniently, efficiently, and securely?

- What you see in the following is a non-prioritized list of our discussions and an attempt to summarize it following the charge questions for the retreat.

What we do now - a short overview

- HEP computing is based on distributed High Throughput Computing (dHTC)
  - We record and store data longterm
  - We sequentially process data using distributed compute resources, disk caches local and detached and strong wide-area-networks
  - We analyze the output of central processing, the more data, the less interactive the analysis process

- HEP software is based on homogeneity both in underlying architecture and language
  - Software frameworks are written in C++
  - Analysis software uses ROOT either through C++ or Python
  - Community solutions are on the rise and very successful ➜ cross-cutting multi-experiment
  - We are through mostly using software that was developed for the HEP community by the HEP community
  - Simulation is based on community tools (Pythia, Geant, …)

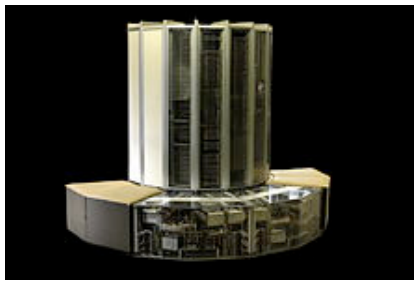
- HEP is developing and using software and computing as a community
  - We share resources across administrative boundaries
  - We develop code in large teams of physicists and computing professionals
  - We protect our data and resources from unwanted access

- We are very successful in doing this and need to continue provide these in the short and near term future, and evolve this for the long term future to stay equally successful

Tomorrow's facilities

- What's old is new again...

Centralized Mainframes & Vector Machines



Harvard

Yale

Enormous State Univ.

HPC Facilities

[Commercial] Cloud Facilities

Yesterday Today Tomorrow

Fermilab

What the future holds

- Working Hypothesis: The future of HEP software and computing will look very different than it does today

- We are no longer the biggest fish in the pond
  - In scale ➜ Industry surpassed us manyfold in data volume, processing capacity, analysis activity
  - In software solutions ➜ Industry and the open source community are using better optimized solutions for orchestration and analysis with a much broader developer and user base
  - In use cases ➜ Other science disciplines are catching up in their dHTC and HPC needs

- The technology landscape is shifting
  - Computing Hardware advances are less and less commercially viable and therefore slowing down
    - End of frequency scaling with Moore's law, emerging of highly parallel architectures ➜ parallelism both at SIMD/vectorization and thread level
    - Hard drive areal density improvements have slowed
  - Heterogeneous systems are coming ➜ CPU/MIC/GPU/FPGA/… bringing also heterogeneity in OS and software architecture
  - Administrative boundaries get expanded to commercial clouds/supercomputer centers ➜ authentication and authorization paradigms get challenged

- The HEP software paradigms are getting challenged across the board
  - Developing HEP software needs more and more specialized CS skills
  - Deep Learning is challenging our way of developing algorithms to extract physics meaning
  - We no longer own the open source data analysis domain. The goal is to have a conversation with your data and allow to fail quickly (interactivity) ➜ independent of the data volume scale
  - How could we use advanced paradigms like quantum computing?

Addressing the charge in one slide

- Draft a list of possible long-term lab goals (not prioritized)
  - Move the field into the era of heterogeneous hardware architectures and enabling software technologies
  - Continue the community approach to solutions but de-emphasize HEP-centric solutions ➜ work closely with the Computer Science Community (“partner with them to solve interesting problems”)
  - Data volume will be the biggest challenge for the future ➜ Solve the exascale data storage, access for processing and analysis problem.
  - Develop exploration capabilities at the lab, develop into a hub for technology and hardware advancements
    - Including integrating and motivating the experiments/collaborations to be part of the revolution
  - Working together in large groups spanning more disciplines and administrative boundaries ➜ Secure federated approaches will be needed
  - Change the physics software development paradigm, enable physicists to write algorithms in a high level language, auto generate optimized code for heterogeneous hardware solutions
  - Return interactivity for the analysis process, no matter the data volume.
- Outline any input and/or additional work that would be required to prioritize each goal
  - A community approach to estimate computing and software needs of experiments/collaborations is needed to expose the true cost of enabling science.
- Provide a rough summary of any upgrades to the accelerator complex or lab facilities required to accomplish these goals.
  - Advanced compute hardware (GPU, MIC, FPGA) and storage hardware (object stores, fast analysis facilities) are needed to develop the capabilities needed for the exascale
  - The gap between CS experts and Physics experts is widening, both cannot be covered anymore by the same people ➜ we need to invest in local expertise to facilitate
    - Train and employ the next-to-next HEP computing experts
  - Facilitate Deep learning by optimizing storage for non-sequential access
  - Community physics code (e.g. event generators) need to be able to take advantage of the parallelism models at scale on modern hardware platforms

Appendix

- Divider: what follows is an abbreviated list of ideas, there is more in the google docs: http://tinyurl.com/kjjrubh

Ideas

- Modeling and simulating computing needs are important for planning. We need toolkits to make it easier for groups to plan - all the way from the trigger level through HTC production workflows.
- We should invest effort to make the full set of costs associated with computing easy for experiments to understand for improved resource allocation decision making.
- Collaborations are globally distributed and need to work across a variety of administrative boundaries and time zones. We must critically review systems to make global collaboration efficient and safe. We should also explore and support new communications technologies.

Ideas

- Computing has grown in complexity rapidly over the past decade, and this trend is accelerating. It is not possible anymore to ask students and young scientists working on an experiment to understand both the physics they are working on and their full computing model, or how to program in all the required systems.
  - We may benefit from framework design that allows users to express algorithms at a very high level, with automatic code generation and optimization happening at a lower level, factorizing the required expertise.
  - Portability of code across a variety of architectures and parallelization strategies is extremely challenging. We need to be able to work with algorithms interactively with small data sets and also seamlessly scale to gigantic data sets efficiently.

Ideas

- We should study the sociology of software management and try to better map the real working conditions of HEP experiments to our planning process (we are more of federation of volunteers than a corporate entity).
- HPC and cloud facilities are likely to provide the bulk of our production computation needs in the future. We will need to redesign our workflows to accommodate this change. But continuous investment in local computing resources is required for actual analysis, which we want to be as close to an interactive experience as possible. We need investment and research into producing and maintaining facilities capable of supporting this.

Ideas

- Programming and algorithm design skills are essential for the modern physicist and this must be reflected in the training and curriculum during education.
- There has been an explosion of new tools and ecosystems for interactive, reproducible, sharable, and visual analysis tools. Enormous active communities are pushing the performance and usefulness of many of these tools (e.g., Jupyter and the scientific Python stack). We need to understand how we can better leverage these tools and the investments being made in them.
- Simulation with quantum computing is a possible avenue for investment and a way to leverage the enormous potential of that technology.

Ideas

- Simulation performance is a major challenge. We expect orders of magnitude larger data sets in the future and simulation needs historically outpace detector data. We need to invest in simulation performance using a multitude of strategies (no one improvement will get us what we need).
- Event generators must also be written to run at large scale and we need to invest effort (especially on the neutrino side) in building more effective ties between the generator and theory communities so we may better understand how to present and execute calculations.

Ideas

- We must be able to easily handle non-standard workflows to exploit new computing architectures to the maximum degree. Where this isn't practical, portable solutions and automatic code generation that is architecture aware will be important.
- We need to understand the sorts of problems where machine learning can add value. We also need to better understand how to effectively input physics we know into algorithms so they don't waste cycles rediscovering effects we understand well.
- Our entire software development model may need to be revisited. Centralized work gives higher quality code and consistent data products, but distributed work provides more resources and is closer to the analysis. We must balance the two.

Ideas

- We need close integration of trust federations into our infrastructure in order to scale security solutions.
- We need to be able to function in a landscape where more diverse resource providers operate through cloud gateways. Commercial clouds and HPC facilities all have fundamentally different access and control models, and they are all different from the traditional HEP grid environment.

Ideas

- We have enjoyed a long period of computing resource homogeneity, but resource heterogeneity is here. Data access patterns have followed a fundamentally sequential paradigm, but newer analysis techniques (e.g., deep learning) are very inefficient in a sequential paradigm.
- Shifting national cyber-infrastructure priorities mean we must be able to operate on centralized "leadership class" supercomputers that dwarf dedicated HEP computing resources.
- Scaling laws are ending across the board. CPU feature scaling has slowed considerably and hard drive areal density improvements have nearly vanished. Competition has diminished across all sectors of hardware manufacturing.

Ideas

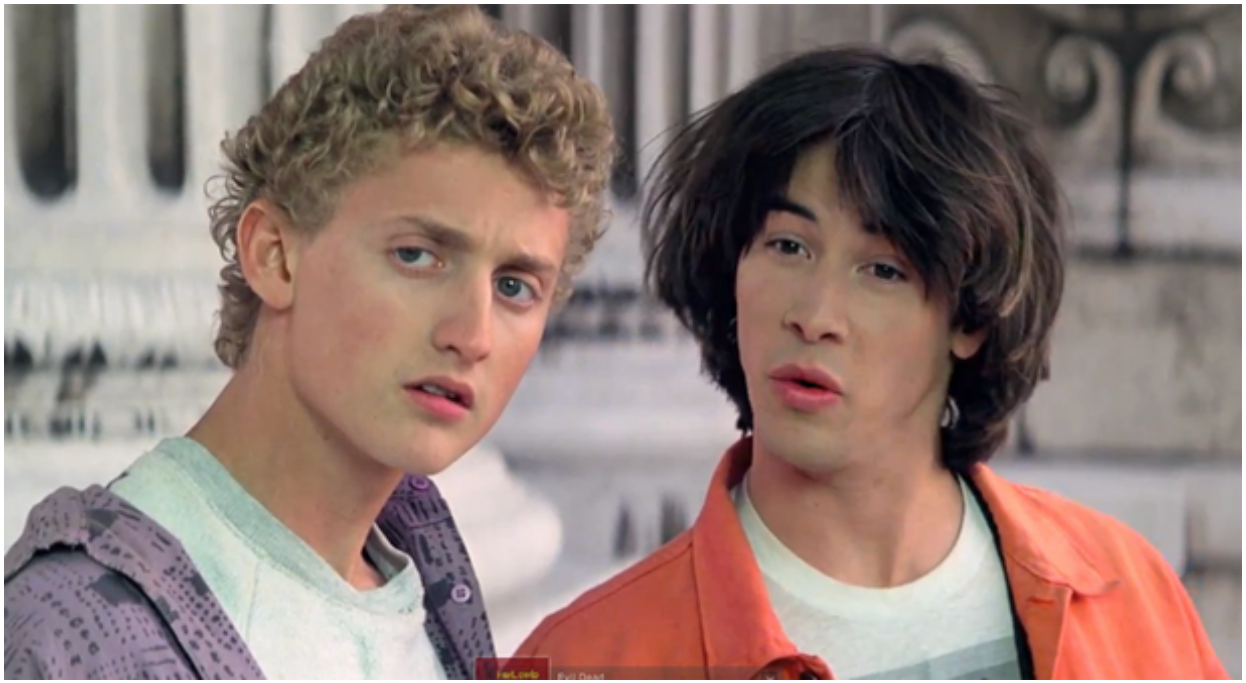
- The era of "one size fits all" computing is over. Can we optimize a set of facilities for new analysis techniques?
- Software is moving away from monolithic "home brew" and experiment specific stacks operating on serial data. It is important to be modular and reusable across many experiments and we must exploit parallelism everywhere. We are about to enter an era where computation as a resource will be abstracted into a nearly invisible service layer and we need to align with computing trends to get the resources we require.
- Filesystems will migrate to abstract stores and data management must work at exascale. "Ntuple" analysis strategies break down if the backend storage does not scale or distribute out.

Ideas

- Tomorrow's frameworks must hide the complexity of environment and facility. Managing those details in analysis code is impossible.
- We will require location-agnostic workflows and smart submission and resource mapping. Users will have to move into a regime where they don't know "how" data is being analyzed, but instead they must properly specify the set of abstract transforms they require. We must think in terms of events and not files.

Ideas

- Quantum computing might provide an amazing number of FLOPs, but we currently have no clue how to use it.



Ideas

- We need to understand and explore the programming abstractions that best position us to fully utilize tomorrow's computing platforms.
- Fundamentally, an analysis framework must provide a simple API to describe analysis without explicitly specifying the parallelism model and it must perform and scale well.
- We need to be able to use modern high performance file formats.