Benchmark Modeling and Projections

Summit and Sierra Supercomputers - From Proposal To Acceptance

Jaime H Moreno
IBM Thomas J. Watson Research Center

Current #1 and #2 systems in Top500 list
https://www.top500.org/

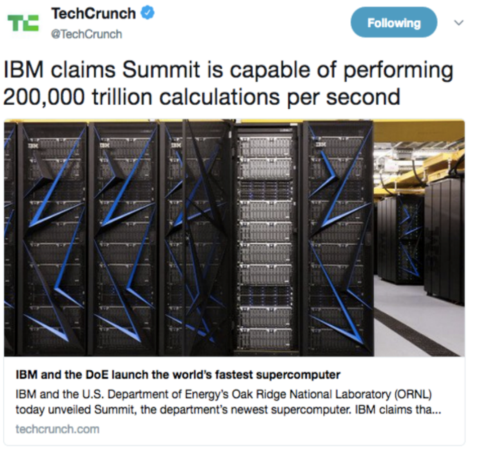
Acknowledgements

- Multi-year efforts by team of experts from IBM and NVIDIA, with support from Mellanox
- IBM
  - Bilge Acun, David Appelhans, Daniele Buono, Constantinos Evangelinos, Gordon Fossum, Nestor Gonzales, Leopold Grinberg, Apo Kayi, Bryan Rosenburg, Robert Walkup, Hui-fang (Sophia) Wen, James Sexton
- NVIDIA
  - Steve Abbott, Michael Katz, Jeff Larkin, Justin Luitjens, Steve Rennich, G. Thomas-Collignon, Peng Wang, Cyril Zeller

- Support from Summit and Sierra System Administrators
  - Summit: Veronica Vergara (ORNL), Jason Renner (IBM)
  - Sierra: Adam Bertsch (LLNL), Sean McCombe (IBM)
- Hardware and Software development and deployment teams across IBM, NVIDIA, Mellanox

- Paper “Benchmarking Summit and Sierra Supercomputers: From Proposal to Acceptance,” *6th Special HPCS Session on High Performance Computing Benchmarking and Optimization* (HPBench 2019).

June 2018: Fastest Supercomputer in the World



TechCrunch
@TechCrunch

IBM claims Summit is capable of performing 200,000 trillion calculations per second

IBM and the DoE launch the world's fastest supercomputer
IBM and the U.S. Department of Energy's Oak Ridge National Laboratory (ORNL) today unveiled Summit, the department's newest supercomputer. IBM claims tha...
techcrunch.com

CNBC
@CNBC

IBM CEO: World's fastest, smartest supercomputer one of our greatest achievements

IBM CEO: World's fastest, smartest supercomputer one of our greatest achie...
The Department of Energy partnered with IBM and Nvidia to deliver the world's fastest supercomputer, called Summit.
cnbc.com

CNET News
@CNETNews

It's as big as two tennis courts and has 9,216 processors boosted with 27,648 graphics chips.

IBM's gargantuan Summit is the 'world's smartest' supercomputer. We'll soo...
It's as big as two tennis courts and has 9,216 processors boosted with 27,648 graphics chips.
cnet.com

Current #1 and #2 systems in Top500 list
https://www.top500.org/

WIRED
@WIRED

America hasn't possessed the world's most powerful supercomputer since June 2013, but the unveiling of the IBM-built Summit supercomputer just changed all of that

The US Again Has World's Most Powerful Supercomputer
A new computer at Oak Ridge National Lab can perform 200 quadrillion calculations per second, ending China's reign.
wired.com

Financial Times
@FinancialTimes

An IBM-designed US supercomputer is set to become the world's most powerful

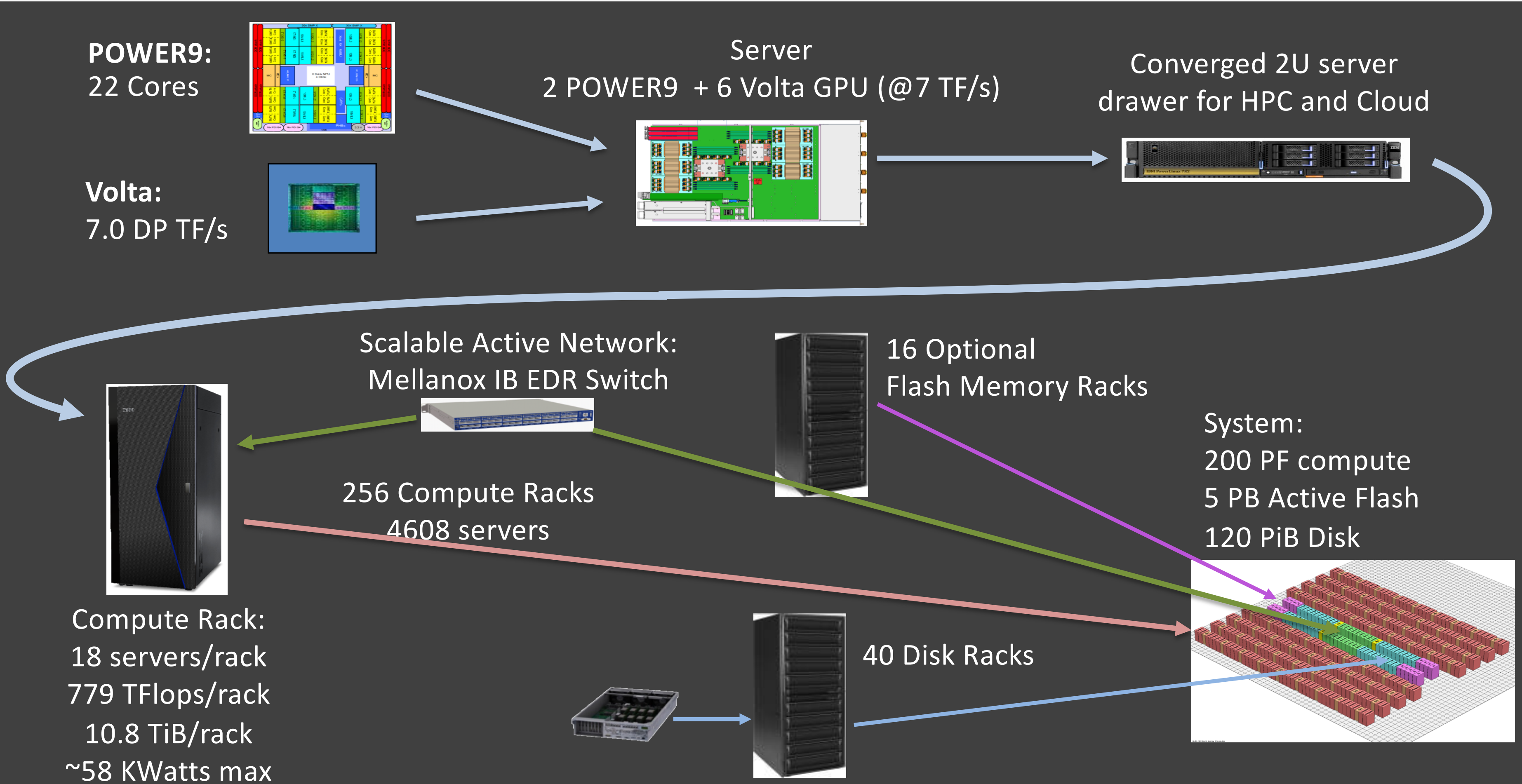
IBM builds world's most powerful supercomputer to crack AI
Summit machine boasts 200 petaflops and was designed with big data in mind
ft.com

The New York Times
@nytimes

The U.S. just beat out China to develop the world's fastest supercomputer. You'd need 6.3 billion years to match what it does in one second.

Move Over, China: U.S. Has the World's Fastest Supercomputer Again
For the past five years, China has had the world's speediest computer. But as of Friday, Summit, a machine built in the United States, is taking the lead.
nytimes.com

Summit's structure



POWER9:
22 Cores

Volta:
7.0 DP TF/s

Server
2 POWER9 + 6 Volta GPU (@7 TF/s)

Converged 2U server
drawer for HPC and Cloud

Scalable Active Network:
Mellanox IB EDR Switch

16 Optional
Flash Memory Racks

System:
200 PF compute
5 PB Active Flash
120 PiB Disk

256 Compute Racks
4608 servers

Compute Rack:
18 servers/rack
779 TFlops/rack
10.8 TiB/rack
~58 KWatts max

40 Disk Racks

Benchmark Modeling and Projections: from Proposal to System Acceptance

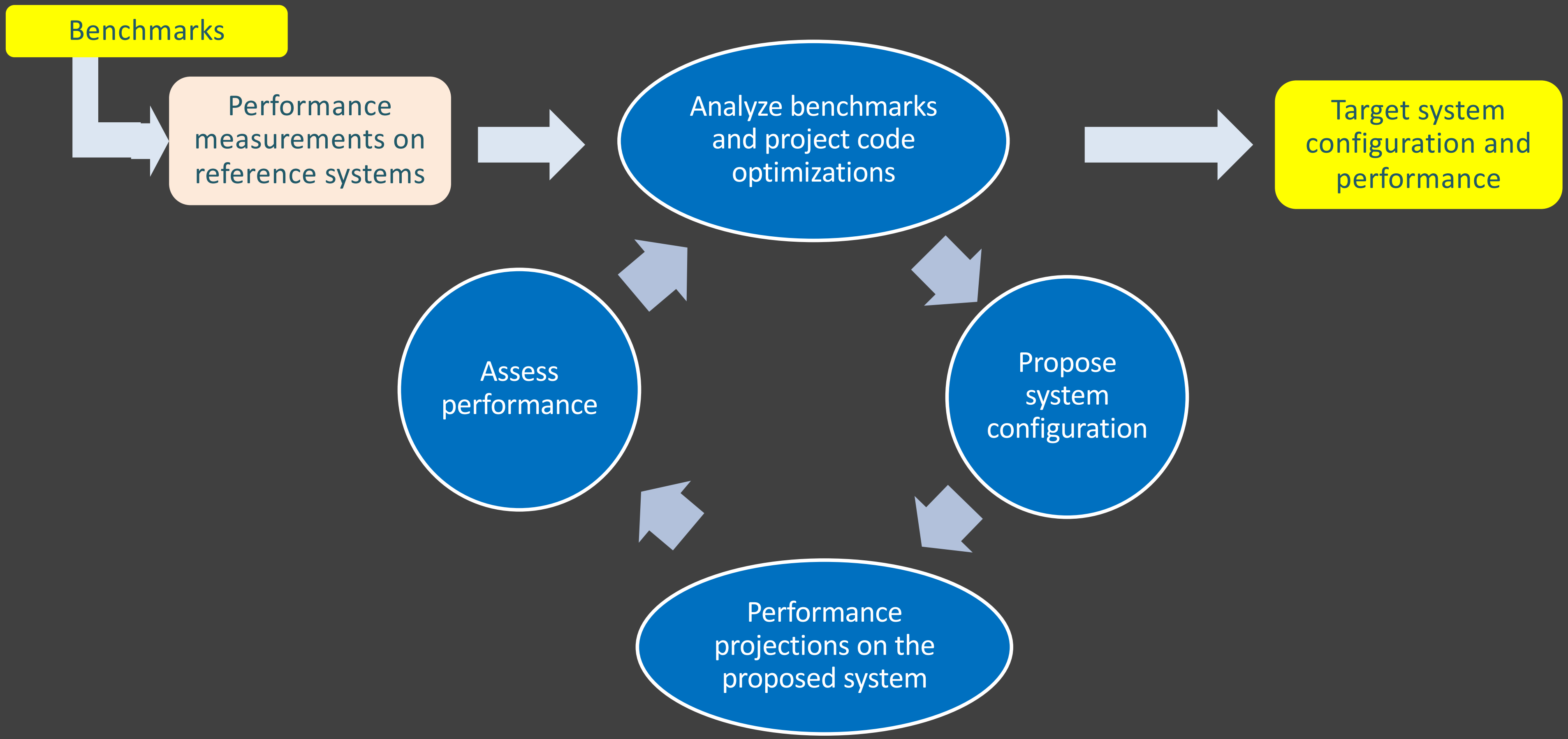
- “Art of Benchmarking” for to-be-developed supercomputers
  - First-of-a-kind systems
  - Modeling and projecting performance while making many hardware and software assumptions
  - Leverage existing systems to extrapolate for future systems
  - Limited ability to perform simulation of future systems

- Very different from benchmarking already deployed supercomputers
  - Different objectives, different methodologies, etc.

- Salient attributes of this process
  - Predefined set of benchmarks representing the target applications, defined by the requester
  - Stringent process to make reasonable yet aggressive projections assuming new architectures
  - A great opportunity for co-design process
    - From initial proposed system’s specification and attributes
    - To the refinement of systems and the design of the entire software stack
  - Validate system’s specifications with respect to expected and contractual attributes of the system

- Note: Procurement of other large scale systems sometimes exhibits somewhat similar characteristics

Co-Design Iterative Process: Initial phase



Benchmarks

Performance measurements on reference systems

Analyze benchmarks and project code optimizations

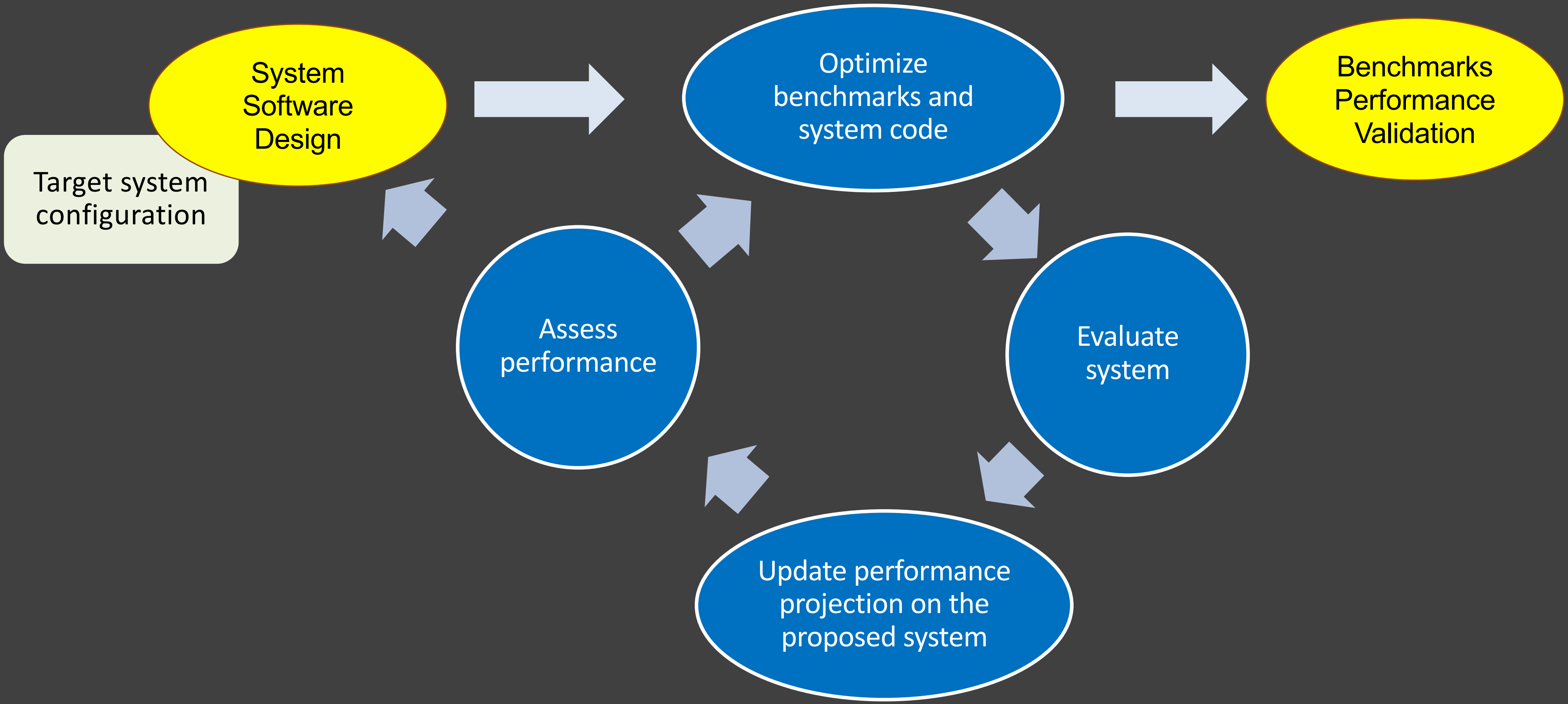
Target system configuration and performance

Propose system configuration

Performance projections on the proposed system

Assess performance

Co-Design Iterative Process: Development and Deployment phase



System Software Design

Target system configuration

Optimize benchmarks and system code

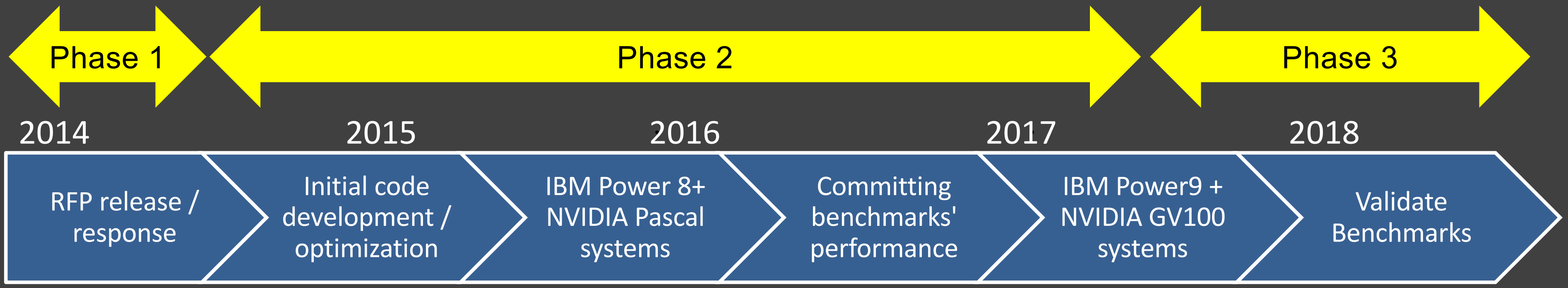
Benchmarks Performance Validation

Assess performance

Evaluate system

Update performance projection on the proposed system

Summit and Sierra: Timeline from Proposal to Acceptance



- Original projections
- Access to DOE systems (Titan, BG/Q)
- Projections = “Targets”

- Code development for new architecture
- Interlock with system software team (e.g., IBM XL and LLVM compiler)
- Access to IBM S822LC (POWER8+P100)

- ”GO/NOGO” checkpoint
- Access to IBM internal cluster with early POWER9+GV100
- Projections = “Committed Targets”

- Ongoing tuning and optimization work
- Benchmarks validation on Summit and Sierra

CORAL Benchmarks: Five Categories

- **Scalable Science Applications**
  - Expected to run at full scale of the CORAL systems (at least 90% of machine) (4600, 4300 nodes)
  - Target **4-8X** improvement for full science runs relative to Sequoia (BGQ)/Titan
- **Throughput Applications**
  - Represent large ensemble runs; run many copies simultaneously (24*192, 20*216) on all nodes
  - Target **6-12x** performance improvement for large ensemble/throughput simulations relative to Sequoia (BGQ)/Titan
- **Data Centric Applications**
  - Represent emerging data intensive workloads
- **Skeleton Applications**
  - Investigate various platform characteristics
- **Micro Applications**
  - Small code fragments that represent expensive compute portions of some of the scalable science and throughput applications

- Figure of Merit (FOM) for each benchmark
- Two variants
  - Baseline: only compiler directives allowed, no code changes
  - Optimized: all types of changes allowed
- Expected performance improvement
  - Geometric mean of FOM ratio over existing reference systems

Benchmarks description - https://asc.llnl.gov/CORAL-benchmarks

	Benchmark	Description	
Scalable Science	LSMS	First principles ground state calculations of solid state systems and statistical physics calculations with a foc **Single node performance with focus on dense linear algebra and parallel scaling efficiency to full system**	Each one scaled to run on the entire system (4600, 4300 nodes)
	QBOX	First-principles molecular dynamics code to compute the properties of materials directly from the underlyir **Parallel dense linear algebra, carried out by the ScaLAPACK library, and a custom 3D Fast Fourier Transfor**	
	HACC	N-body techniques to simulate formation of structure in collisionless fluids under the influence of gravity in **Three distinct phases in the computation: stride-1 vectorizable, irregular indirect with branch and integer**	
	Nekbone	High order, incompressible Navier-Stokes solver based on the spectral element method **Conjugate gradient iterations that call matrix vector multiplication operation in an element-by-element fa**	
Throughput	LULESH	Hydrodynamics stencil calculation using both MPI and OpenMP to achieve parallelism **Compute performance properties more interesting than messaging (only ~10% of runtime spent in comm**	At least 24 jobs running simultaneously, filling up the entire system
	CAM-SE	Atmospheric climate modeling; hydrostatic Euler equations with added multi-scale physics representing clir **Parallel efficiency using a large portion of the target system**	
	QMCPACK	Continuum quantum Monte Carlo simulation; particle positions randomly sampled according to various QM **High weak and strong scaling; ability to optimize C++ template constructs and vectorized math library**	
	NAMD	Classical molecular dynamics code that simulates molecular interactions using Newtonian laws of motion **Object-oriented style using the asynchronous data-driven language Charm++**	
	AMG	Algebraic multigrid solver for linear systems arising from problems on unstructured grids **Single CPU performance and parallel scaling efficiency; very large demands on main memory bandwidth**	
	UMT	Three-dimensional, non-linear, radiation transport calculations using deterministic (Sn) methods **Combination of message passing and threading, large distributed memory, unprecedented (weak) scaling**	
	MCB	Monte Carlo particle transport benchmark **MPI+OpenMP parallel scaling efficiency; branching and integer computations**	
	SNAP	Spatially 3-D, time-dependent calculation using discrete ordinates. Mimics workflow/communication patter **Stresses memory subsystem and total memory capacity**	

FOM: Representing a rate of execution

Total Time

$$T_{total}(s) = \frac{T_{k20x}(s)}{V} + Tcopy(s) + Tmpi(s) + Tcpu(s)$$

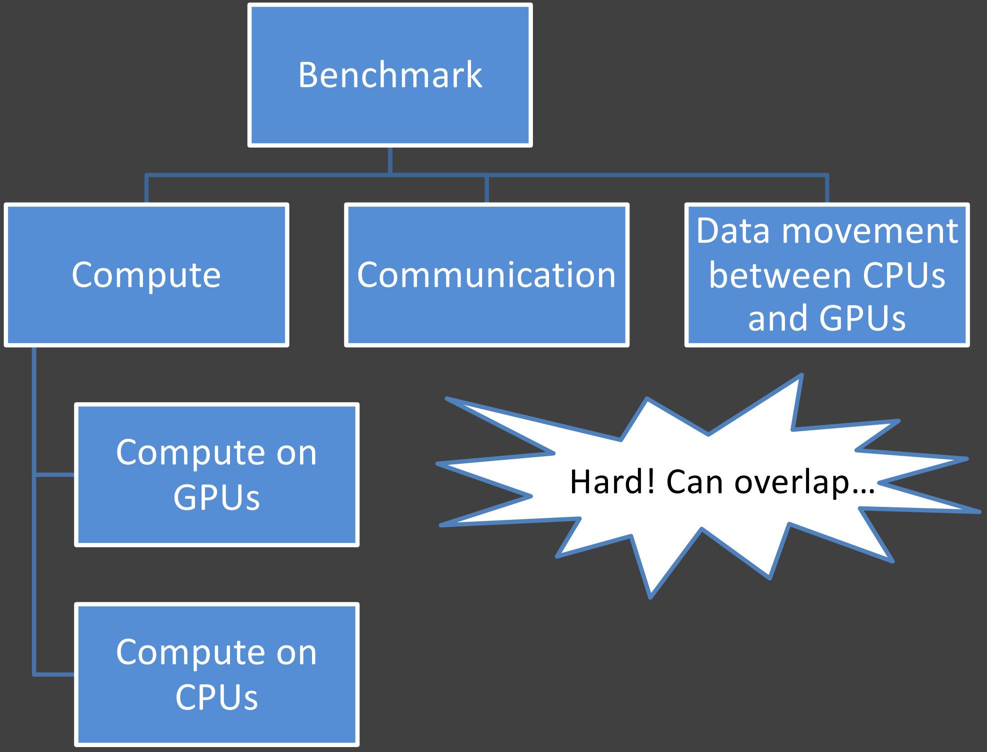
Figure-of-metrit (FOM): zones/second

$FOM = \frac{p \times s^3}{T_{total}(s) \times 1000}$ where $p \times s^3$ is the total number of elements

Speed-up relative to the reference

$S_i = {}^{FOM}/_{1.118E+07}$ where $1.118E + 07$ is the FOM on the reference system

Phase 1: Projecting Performance



Benchmark

Compute

Communication

Data movement between CPUs and GPUs

Compute on GPUs

Compute on CPUs

Hard! Can overlap…

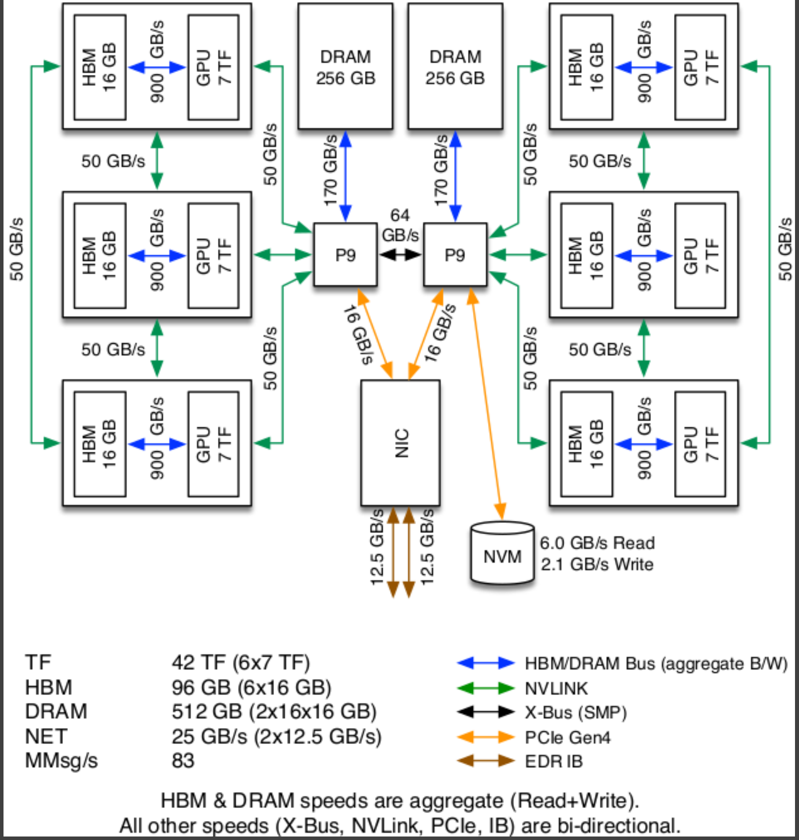
1. Benchmarks characterization in BGQ and Titan
2. CPU-only projections
   - POWER7 measurements scaled (ratios: bandwidth, SPECfp, ...)
3. GPU acceleration
   - Kernels ported, K20/K40 measurements scaled (ratios: bandwidth, SMs, memory, flops, ...)
4. Parallel efficiency at scale (4600+ nodes)

- Total Time = CPU + GPU + MPI + Data Movement
  - Worth moving computation to GPUs?
- Compute kernels
  - Flops, memory or latency bound
- Compiler maturity
  - For directive-based approach, only OpenACC was available initially

- Performance projections = “Targets”

Phase 1: Attributes/Specifications of the Proposed System

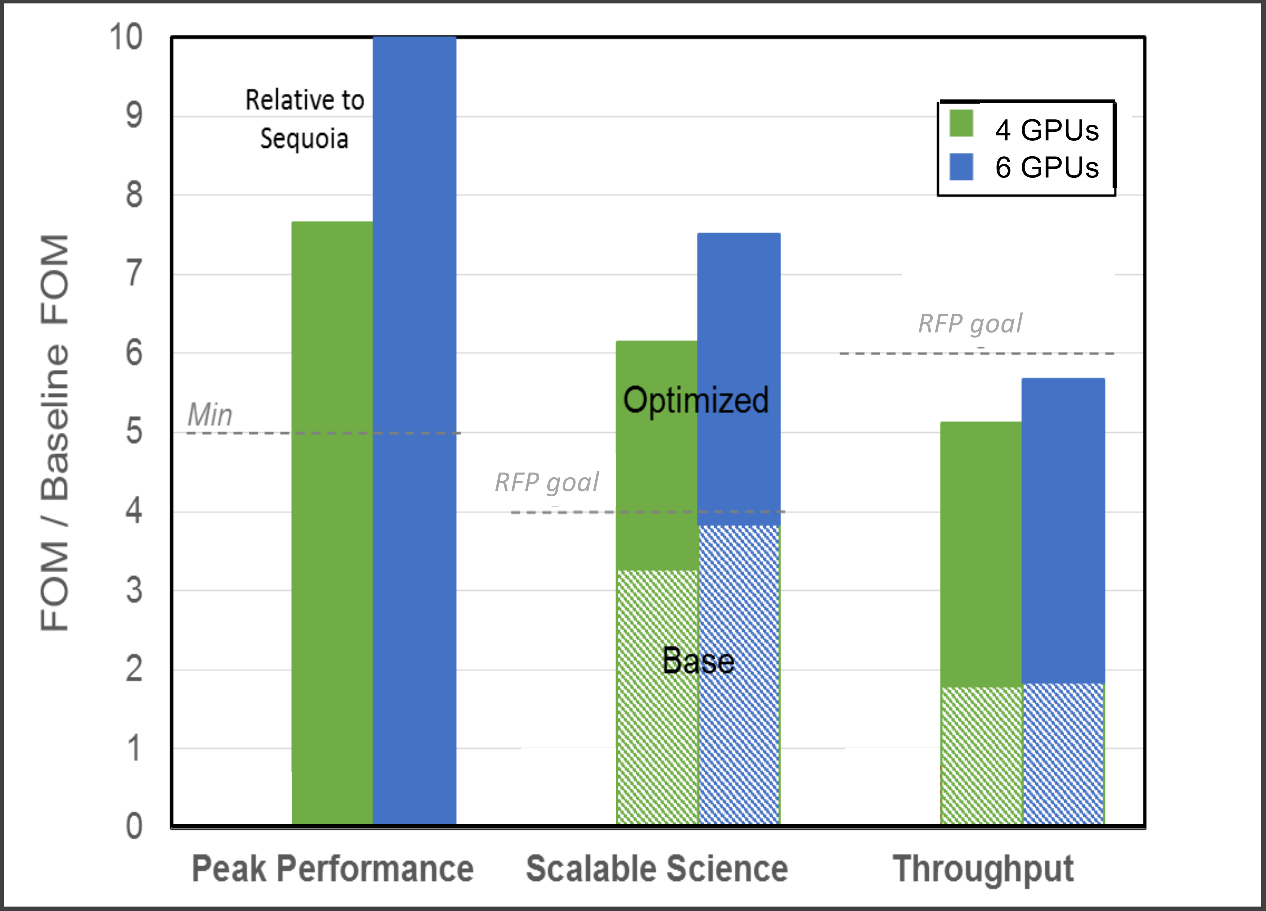
- Node count
  - Solve target CORAL problem size -> problem size per node
    - Off-node MPI data volume
    - Data transfer via NVLinks
- Interconnect: network capabilities
  - MPI collective calls performance
  - Time spent in message exchange
- Attributes of the compute engines
  - Peak flops
  - Peak/Sustained memory bandwidth
  - Cores/SM counts (shared memory, register file, etc)
  - Sizes of Caches
  - Speedup scaling factors: CPU, GPU, network
  - More…
- Assumptions, such as
  - "OpenACC no worse than 3x CUDA"
  - ….
  - ….



TF	42 TF (6x7 TF)
HBM	96 GB (6x16 GB)
DRAM	512 GB (2x16x16 GB)
NET	25 GB/s (2x12.5 GB/s)
MMsg/s	83

HBM & DRAM speeds are aggregate (Read+Write).
All other speeds (X-Bus, NVLink, PCIe, IB) are bi-directional.

Phase 1: Performance Projection in the RFP Response

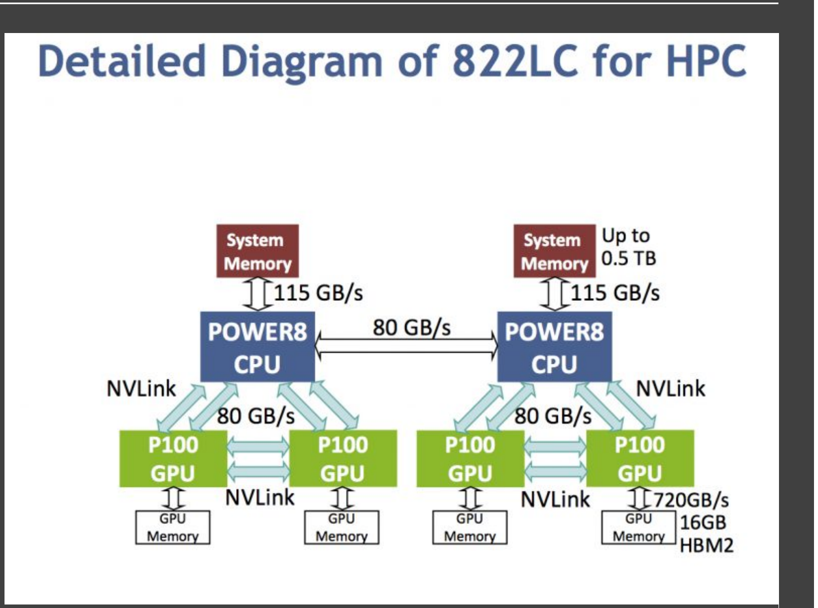


- Baseline: only compiler directives allowed, no code changes
- Optimized: all types of changes allowed

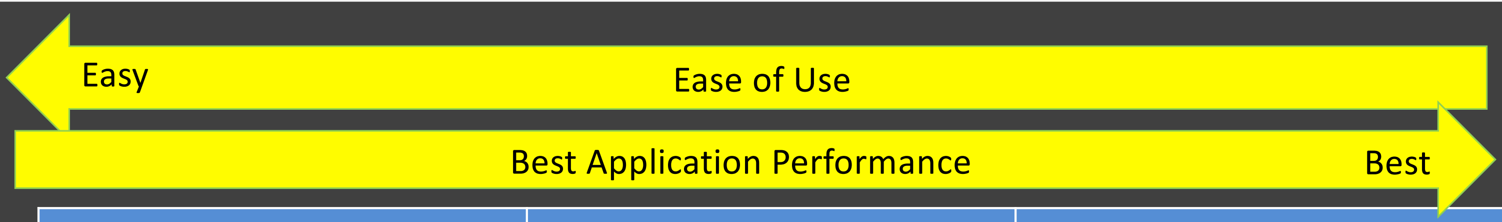
Performance projections became contractual "targets"

Phase 2 – Steps leading to Go/NoGo checkpoint

- Similar process to Phase 1
- Access to POWER8+P100 system (822LC)
  - Measurements and projections updated based on hardware platform closer to target systems’ characteristics
  - Crucial for “Go/NoGo” decision
- Continuous improvement to the codes
  - Move kernels from CPUs to GPU
  - Refactor codes
  - Manage data movement
- Co-design effort
- Simultaneously, IBM and NVIDIA teams assisted DoE labs to ready their applications for the CORAL systems
  - Centers of Excellence (CoE)
  - “Early-Access” systems (POWER8+P100)



Performance Enhancements via GPU Acceleration



Libraries • ESSL/PESSL • NVIDIA Libraries • Math library, cuBlas, NPP, etc	Programing models supporting directives • OpenACC • OpenMP	Programing language targeting GPU • CUDA
• Easy to Implement • Tested and Supported • Limited – needs may not be covered	• Modification of existing programs with directives • Compiler assists with mapping to device	• Most time intensive • Requires expertise • Achieves best performance results

Phase 2 – Go/NoGo checkpoint

- Target system software not yet fully available
  - Compilers, libraries, CUDA, etc.
- Performance projections updated with revised scaling factors
  - Estimates of expected improvements were included in projections
  - Different estimates given different attributes of systems (eg, 2 or 3 GPUS per CPU)
- A few hardware design changes had been adopted
- System configuration changed, driven by evolution of cost tradeoffs
  - Systems became more different than initially conceived, adding challenges to the benchmarks projections process
  - e.g., DRAM capacity, bi-section network bandwidth
- In spite of these factors, NO changes were made to the projections for both systems
  - Confidence on the projections process being applied and code optimizations in progress

After Go-decision, performance projections became contractual **obligations**

Phase 3: Final systems specifications

	Summit (Oak Ridge)	Sierra (Livermore)
Peak Performance	200 PetaFlops	125 Petaflops
Number of Nodes	4608	4320
Node Performance	43 Teraflops	29 Teraflops
Compute per Node	2 POWER9 6 GV100	2 POWER9 4 GV100
Total Compute	9,216 POWER9 27,648 GV100	8,640 POWER9 17,280 GV100
Memory per Node	512 GiB DDR4 96 GiB HBM2	256 GiB DDR4 64 GiB HBM2
File System - GPFS	250 PiB 2.5 TiB/s	156 PiB 1.5 TiB/s
Power consumption	15 MW	12 MW
Interconnect	Mellanox EDR 100G InfiniBand	
Operating System	RedHat Enterprise Linux (RHEL) 7.4	

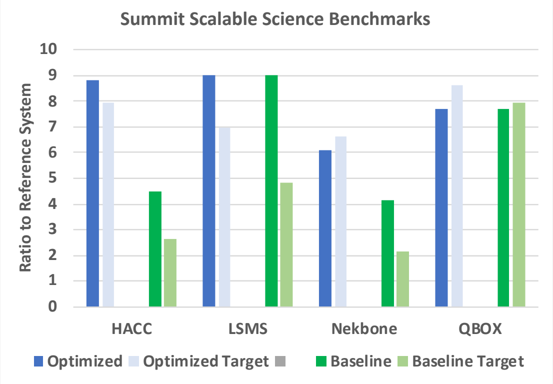
Phase 3: Validation of Projected Performance

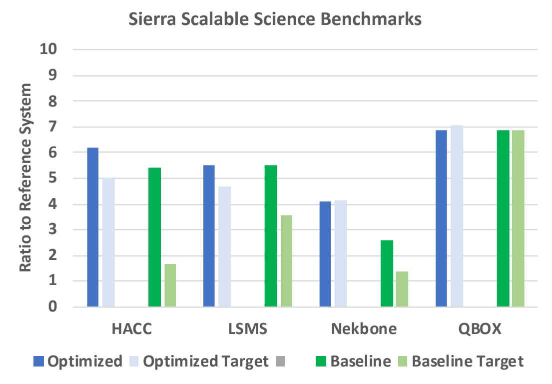
- Mini-CORAL cluster available internally at IBM Research
  - Combination of up to 256 POWER9+GV100 nodes: (4 GPUs / 256 GB), (6 GPUs / 512 GB)
  - Critical to conduct the final tuning and code optimization
  - Validate the quality of software to ensure no performance regression
  - Nonetheless, early-hardware and pre-release software constraints
- Experiments performed on Summit and Sierra as the systems were being brought-up
  - 1/4th system delivered December 2017 (~1024 nodes)
  - Live debug sessions including representatives across the software stack
  - Possible to run throughput benchmarks at scale (~200 nodes)
- Final validation of benchmarks performance on Summit and Sierra
  - Single 4-days period allocated at each site (over long holiday weekend, in one case….)
  - Systems still undergoing final stages of deployment (hardware and software)
  - Issues surfaced at this stage mostly related to scaling code to run at larger scale
    - Long bootstrap/startup time for MPI applications at scale
    - Variability introduced by operating system noise and hardware behavior
    - Random failure in the applications due to the instability with software stack still undergoing development
- Most problems were identified and fixed before entering formal acceptance

Benchmarks performance results

- Measurements collected prior to entering system acceptance , during a 4-days sessions at each site
  Additional system tuning took place afterwards, leading to further improved performance
  - Did not have access to the systems to repeat measurements

Scalable Science


Summit Scalable Science Benchmarks

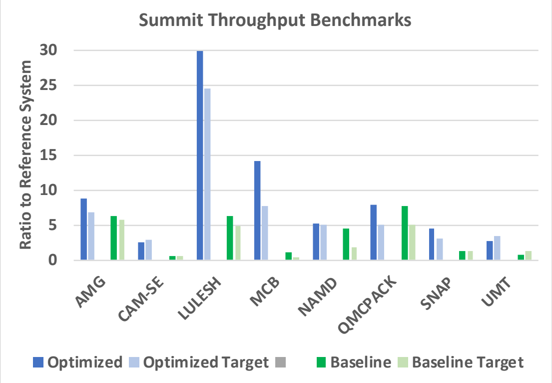

Sierra Scalable Science Benchmarks

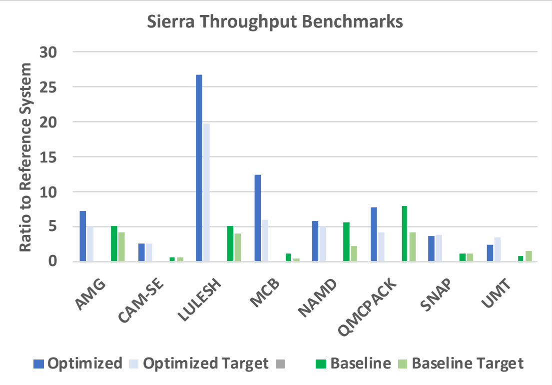
GeoMean Ratio over reference and % to projected value

	Baseline	Optimized
Sierra	4.97 (182%)	5.87 (118%)
Summit	6.00 (156%)	7.84 (1.04%)

RFP Optimized spec: 4x

Throughput


Summit Throughput Benchmarks


Sierra Throughput Benchmarks

	Baseline	Optimized
Sierra	2.36 (132%)	6.40 (125%)
Summit	2.41 (132%)	6.85 (121%)

RFP optimized spec: 6x

Validation Lessons Learned

- Prepare microbenchmarks to measure the health of the system
  - e.g.: CPU clock frequency, sustained memory bandwidth, flops, NVLink bandwidth, network bandwidth, among others.
  - Consolidate data to quickly identify problems (a lot of data, 4600x6 GPUs)
- Select few simple benchmarks that are easy to build and run
  - Validate functionality and performance of new software release, firmware/OS update, etc
  - Oftentimes, micro/skeleton benchmarks are too simple to catch performance regression
  - Testing on few nodes is actually sufficient
- Automatic testing framework (e.g., “harness”) essential
  - Continuously fill up entire system (> 4000+ nodes) with limited users’ intervention
- Maintain historical performance data for key benchmarks
  - MPI profiling data is especially critical
  - Can help narrow down the stability or variability issues within the system
- Work collaboratively and productively among HW/SW/Application teams

Some take away comments

- Regression does happen: performance, functionality
  - Worth spending efforts enhancing the testing suite
- Software design for large scale systems without having access to big systems for testing
  - Need to include scalability in the design from the start
  - Develop capabilities to gather different levels of telemetry to assist debugging at scale
- Benchmarking (and designing) new HPC systems is a very complex process
  - It’s not going to be a smooth process, as all of the pieces are moving targets
  - New system architecturally different from prior systems, making projections a difficult task
  - Right set of people/skills working together is crucial
- Performance projection of large scale system is still an “art”
  - Multiple assumptions made early and throughout the process
  - Multiple adjustments required during development
  - Highly dependent on prior expertise
- Opportunity for advancing state of the art towards a more established science of benchmarking large systems while undergoing development

Summary

- Summit and Sierra were delivered to ORNL and LLNL labs, on schedule
- Benchmarks projections made 4 years in advance were exceeded – real achievement..!
  - In spite of multiple challenges throughout development period
- Expecting improved performance as the systems becomes more mature
  - Further improvements in tools and programming practices
- Real speedup on benchmarks and applications
  - 6-8x speed up on benchmarks over reference system on optimized code
  - Even larger benefits already reported on actual scientific and machine learning applications
  - Meaningful performance gains even with just code annotations
- Compelling feedback from scientists using the systems
  - Videos with opinions by the scientists available on-line
  - Publications and awards
- Benchmark modeling and projections were a crucial component throughout the systems development process