Developing for the Cloud:
How to choose the right software and hardware for an Application
(Sample Slides)

Barbara Chapman, HPE

Sanyam Mehta, HPE

Eric Lequiniou, Altair

James Edwards, NCAR

Ayad Jassim, HPE

Lei Huang, Prairie View A & M

Introduction

Dr. Barbara Chapman, HPE

Relevance in Current Context (1/2)

- HPC as a Service (HPCaaS) increasingly relevant for HPC users
  - Especially the small to mid-size HPC system users
    - Ease of use – rent a processor and some memory/storage and run jobs
    - Cost effectiveness – no need to pay for and install big supercomputers on-site
    - Access to latest hardware – the latest hardware immediately arrives on cloud
  - And even the big-size HPC system users
    - 'Bursting' to the cloud is an important use-case – users of big system need a place to run jobs when the system is loaded or experiencing a downtime (increasingly relevant in the Exascale era)

Relevance in Current Context (2/2)

- Key HPC verticals increasingly favor the Cloud
  - **Financial Services Industry –** Enhanced compute and shorter turnaround time requirements during taxing times such as increased market volatility
  - **Life Sciences (Pharmaceuticals) –** Investigation of major diseases and drugs in an unpredictable world of newer diseases and rising health challenges
  - **Weather Forecasting –** Increasing demand for highly detailed forecasts in a wide range of industries such as energy, transportation, agriculture, etc.
  - **Finite Element Analysis –** Need to replace costly tests related to crash, safety, manufacturability, durability, etc. with quick and efficient simulation
  - **Computational Fluid Dynamics –** Widely used including automobile, aerospace and aeronautics industries that are making rapid advancements

Goals

- Understand and appreciate the challenges for traditional HPC users as they embrace HPCaaS
  - How to choose the right platform with the right combination of software and hardware
- The presenters and attendees explore two potential solutions
  - Solution #1: HPCaaS-Bench, a basis to compare across vendors and also across various software and hardware options
  - Solution #2: Understand and witness the impact of various software and hardware choices on application performance, portability, energy-efficiency.
- Take-aways: Best Practices for users with regards to choosing software and hardware for their applications in HPCaaS

Outline

Session 1: **Introducing HPCaaS-Bench** (30 minutes)
Presenter: **Barbara Chapman (HPE)**
We introduce HPCaaS-Bench, a benchmark suite comprising applications from key verticals in HPCaaS space. We discuss the relevance of these applications, and their specific characteristics in terms of compute, memory, communication, IO bound.

Session 2: **Presenter: Sanyam Mehta (HPE)**
Session 2A - **Create an account and log into HPE's development cloud** (30 minutes)
This session will guide the attendees through creating a user account and log into HPE's development cloud. The users will be able to appreciate the variety of software and hardware available on cloud development platforms.

Session 2B – **Choosing the right processor to minimize cost** (30 minutes)
We use the Monte Carlo Options Pricing application from Financial Services Industry. A real-world example demonstrates that the best processor choice depends not only on processor's throughput but also its rental cost.

Session 2C – **Choosing the right compiler options and parallelizing strategy for a given target** (30 minutes)
Using NAMD from the pharmaceuticals (Life Sciences) industry, the attendees learn the impact of choosing appropriate compiler options and parallelizing strategy (MPI/OpenMP/Hybrid) on application performance.

Session 3 – **Maximizing parallel performance through the right combination of software and hardware** (1 hour)
Presenter: **Eric Lequiniou (Altair)**
We use the communication and IO intensive Altair's OpenRadioss application to highlight the importance of choosing the correct combination of MPI implementation and network interconnect from multiple possibilities. We further show the benefit from using a parallel file system to such applications.

Session 4 – **Choosing among the available compilers for an Application** (45 minutes)
Presenter: **James Edwards (NCAR)**
We use MPAS to represent the weather vertical and show the impact of using different compilers and compiler options on compilation and performance.

Session 5 – **How to best distribute a parallel job across nodes, sockets and cores** (45 minutes)
Presenter: **Ayad Jassim (HPE)**
We use the popular OpenFOAM application to demonstrate the impact of MPI rank distribution on performance.

Session 6 – **Optimizing for power and energy efficiency (and cost?)** (60 minutes)
Presenter: **Lei Huang (Prairie View A&M, Texas)**

We use an Oil and Gas application to demonstrate ways to save power while minimizing performance loss and thereby maximizing energy savings for sustainable computation.

Session 1: Introducing HPCaaS-Bench

Dr. Barbara Chapman

The Two Cloud Paradigms Relevant for HPC

- Platform as a Service (PaaS)
  - Users develop applications in the Cloud
  - Cloud provider provides software (such as compiler, libraries, etc.)
  - User picks and chooses from the available choices

- Software as a Service (SaaS)
  - Users provide data
  - Cloud provider provides pre-built software that consumes user data and runs in a more and less opaque manner on Cloud resources

Challenges Facing HPC Users in the Cloud

- For PaaS users:
  - Many software and hardware options (from various vendors or even with one vendor) to choose from
  - Hard to find the right combination
- For SaaS users:
  - Many vendors provide the required pre-built software
  - Which one uses the right software, hardware under the hood to provide best performance
- Ultimate Challenge is to reduce time to solution
  - Remember that in the cloud, Time = Money, Literally!

Solution #1: HPCaaS-Bench

- A standard benchmark suite comprising representative applications from key HPCaaS verticals

- How does it help?
  - Provides basis to compare across vendors – especially useful for SaaS users
  - Helps to compare across various software and hardware options – especially useful for PaaS users

Applications in HPCaaS-Bench

- Financial Services Industry – Monte Carlo Options Pricing
- Life Sciences (Pharmaceuticals) – NAMD
- Finite Element Analysis – OpenRadioss
- Computational Fluid Dynamics – OpenFOAM
- Weather Forecasting – MPAS-A
- Oil and Gas Industry – Specfem3D

HPCaaS-Bench offers a variety

- Monte Carlo Options Pricing – Single node only; useful for processor performance and price-performance exploration
- NAMD – Very sensitive to compiler options that deal with vectorization. Useful for exploring MPI+OpenMP hybrid performance
- OpenRadioss – Communication intensive, IO intensive; helpful to study MPI performance across implementations, interconnects
- OpenFOAM – Scales well with nodes; useful to study the impact of rank distribution on performance
- MPAS – Sensitive to compilers and compiler optimization flags
- Specfem3D – We study ways to minimize energy with this application

The Remainder of the Tutorial…

- Study the impact of various combinations of hardware and software choices on performance of HPCaaS-Bench applications, and in some cases portability and energy.
  - Hardware choices include processors, interconnects
  - Software choices include compilers, compiler flags, MPI implementation, communication framework, file system, operating frequency, etc.

Session 2A: Understanding the Test System

Dr. Sanyam Mehta

Breckenridge, HPE devCloud

- Variety of hardware
  - AMD Milan nodes with 64 cores per socket, 2 sockets per node
  - AMD Genoa nodes with 96 cores per socket, 2 sockets per node
  - Intel Sapphire Rapids with 56 cores per socket, 2 sockets per node
- Variety of software
  - Cray, Intel, GNU (and other) compilers
  - Cray-mpich, OpenMPI MPI libraries
  - NFS, Lustre file systems
  - Ofi, ucx communication framework

Lab #1: Logging into HPE devCloud

- All attendees get guest accounts for logging in
  - Users create account here: https://breckenridge.cloud/
  - Users log in using simple instructions here: https://breckenridge.cloud/docs/devcloud/quickstart.html
  - Presenters lead attendees in following investigations for an exciting hands-on experience

Session 2B: Choosing the Right Processor to Minimize Cost

Dr. Sanyam Mehta

Finance and Monte Carlo Options Pricing

- Market risk management is critical to trading firms
  - how various combinations of market price movement affect their investments in various financial instruments
- Many financial instruments such as equities, bonds, interest rates, currencies, etc. incorporate options, where the number of options contracts every year are tens of billions
- Monte Carlo method is very popular and relevant to provide the overall most likely option price among the paths an asset's price and volatility could take

Monte Carlo Application

- We use implementation from FinanceBench from University of Delaware (Cavazos et al. [2013])

- We make one change to source code for our evaluations
  - rand() is not thread-safe
  - rand() changed to rand_r() for effective scalability

- Number of samples: 10000000

- Goal of the Lab: Minimize Overall Cost When Choosing Processor

Lab #2

- Attendees are offered a choice between two of the latest processors
  - AMD Genoa with 192 cores per node
  - Intel Sapphire Rapids with 112 cores per node
- Attendees are also provided the price of renting either processor
- Based on observed performance throughput, attendees figure out the better choice!

Sample Results (from our tests)

genoa	target=genoa 1-thread	target=genoa 192 threads	target=genoa 384 threads
	30.1s	172ms (175x)	182ms (165x)
spr	target=spr 1-thread	target=spr 112 threads	target=spr 224 threads
	28.3s	383ms (74x)	346ms (82x)

Execution time of Monte-Carlo with different configurations

	Relative Performance	Relative Price	Performance per dollar
genoa	1	$\frac{\$BM.Standard.E4.128}{128}$ x 192 = $\frac{\$4666.37}{128}$ x 192 = $7000	1
spr	$\frac{162}{327}$ = 0.5	$\frac{\$BM.Standard3.64}{76}$ x 112 = $\frac{\$3047.42}{76}$ x 112 = $4491	$\frac{\$7000}{\$4491}$ x 0.5 = 0.78

Performance per dollar comparison between sapphire-rapids and genoa on Monte-Carlo

Takeaways (from our tests)

- We first observe that Intel sapphire-rapids is 1.06x better than AMD genoa in single-thread performance.
- Genoa scales better than sapphire-rapids
  - This is because clock rate drops more significantly for sapphire-rapids under load.
- We find that for genoa, the best performance is with using 1 thread per core and using all the cores on the node. On the other hand, for sapphire-rapids, the best performance is seen when using both hyperthreads on all cores.
- We find that genoa scores 1 versus sapphire-rapids' 0.78, thus becoming the preferred choice for a user targeting better performance for a given price even though sapphire-rapids beats genoa in single-thread performance.

Session 2C: **Choosing the right compiler options and parallelizing strategy for a given target**

Dr. Sanyam Mehta

Life Sciences and NAMD

- The life sciences industry comprises companies that perform research, development and manufacturing of pharmaceuticals, biotechnology based food and medicines, medical devices, biomedical technologies, food processing, and other products with the aim of improving the lives of organisms
- With ever emerging newer diseases and rising health challenges among population at large, this industry is experiencing steep growth.
  - The global pharmaceutical industry made an aggregate estimated business worth $1.3 trillion in the pre-covid year 2019
- Molecular biology is increasingly relevant to pharmaceutical sciences as it helps in the understanding of structures, functions, and internal controls within individual cells, which can then be used to efficiently target new drugs, diagnose disease, and better understand cell physiology.
- NAMD is a very popular molecular dynamics simulation software and plays an important role in modern molecular biology

NAMD Application

- NAMD is designed for high performance simulations of large biomolecular systems on parallel computers. Written using the Charm++ parallel programming model for parallel efficiency.

- We use the STMV (virus) benchmark with 1,066,628 atoms
  - Known to scale well even up to thousands of processors

- Goal of the Lab: Maximize single-node performance

Lab #3

- Attendees are again offered a choice between two of the latest processors
  - AMD Genoa with 192 cores per node
  - Intel Sapphire Rapids with 112 cores per node
- Attendees will be allowed to build NAMD source (other libraries will be provided pre-built)
  - Attendees could use different compiler targets with the Cray compiler - ‘cray-x86-rome’ and ‘craype-x86-genoa’ for genoa and ‘craype-broadwell’ and ‘craype-x86-spr’ for sapphire-rapids; this allows to test vectorization at different aggressiveness
  - We further provide two different versions of code – one with `ivdep’ pragmas to further promote vectorization for certain key loops, and another without the `ivdep’
  - Attendees also then test running with different combinations of MPI ranks and OMP threads
- Based on observed performance throughput, attendees figure out the better choice!

Sample Results (from our tests)

Processor = **genoa** 8 ranks x 12 threads	target=**genoa**		target=**rome**	
	with ivdep	**w/o** ivdep	**with** ivdep	**w/o** ivdep
	39.1s	39.5s	33.3s	26.9s
Processor = **spr** 8 ranks x 7 threads	target=**spr**		target=**broadwell**	
	with ivdep	**w/o** ivdep	**with** ivdep	**w/o** ivdep
	62.2s	62.9s	68.9s	55.9s

Execution time of NAMD with different configurations

Ranks x Threads	# Worker Threads	# Communication Threads	Execution Time
2 x 48	94	2	26.3s
4 x 24	92	4	26.0s
8 x 12	88	8	26.7s
16 x 6	80	16	28.8s
24 x 4	72	24	31.7s

Execution time of NAMD with different number of ranks and threads

Takeaways (from our tests)

- We find that targeting ‘rome’ and running on genoa is much more performant as compared to targeting ‘genoa’ on genoa both with and without ivdep on certain important routines.
- When using ‘target=craype-broadwell’ and running on sapphire-rapids, we make a similar observation as above where ‘target=craype-broadwell’ and not using ivdep is the most performant configuration and beats ‘target=craype-x86-spr’ by as much as 1.13x.
- Overall, we find that the best performing version on genoa (‘target=craype-x86-rome’ without ivdep) is 2.1x better than the best performing version on sapphire-rapids (‘target=craype-x86-broadwell’ without ivdep).
  - The higher clock rate combined with higher core count together lead to the 2.1x speedup for genoa.
- We observe that the best performing configuration (4 x 24) is as much as 1.22x more performant than the worst performing configuration (24 x 4)
  - Best performance does not come from using the maximum possible number of ranks or threads per node.

Session 3: **Maximizing parallel performance through the right combination of software and hardware**

Mr. Eric Lequiniou

Finite Element Analysis and OpenRadioss

- Altair Radioss is a multidisciplinary finite element solver that helps users evaluate and optimize product performance for linear as well as highly nonlinear problems under dynamic loadings.
- OpenRadioss is the publicly available open-source version of Radioss that was very recently released now being increasingly used by researchers and developers
  - Empowers users to tackle challenges associated with rapidly evolving technologies such as autonomous driving and battery development while also improving traditional use cases such as automotive crash and safety, shock and impact analysis, electronic and consumer goods drop testing, and fluid structure interactions

OpenRadioss Application

- OpenRadioss makes use of both OpenMP and MPI parallel structure, leading to industry-leading scalability regarding large, highly nonlinear structural and multiphysics simulations.
- We use the NEON FE (1M11) benchmark model that simulates a frontal crash of a 1996 Chrysler Neon traveling at 50km/h colliding with a rigid wall.
  - This analysis simulates the performance of the Neon (with refined mesh) through the first 80ms of the crash event.
- OpenRadioss is both communication intensive and IO intensive
- Goal of the Lab: Maximize parallel (multi-node) performance by minimizing communication and IO overhead

Lab #4

- Attendees will be offered two different binaries – one linked with cray-mpich and one linked with OpenMPI
- All tests are performed on Genoa with the Slingshot interconnect and ofi communication framework (although we also have sample results for Infiniband HDR200 interconnect and ucx communication framework)
- Attendees run with different number of nodes and capture observed performance with the two binaries
- Attendees also run with input and output files being read from and written to the NFS and Lustre file systems
- Based on observed performance throughput, attendees figure out the best combinations of software and hardware for maximizing overall performance

Sample Results (from our tests)

Ranks x Threads	File system = nfs4		File system = lustre				
	SS+CrayMpich [Column 1]	SS+openMPI [Column 2]	SS+CrayMpich [Column 3]	SS+openMPI [Column 4]	IB+CrayMpich+ucx [Column 5]	IB+openMPI+ucx [Column 6]	IB+CrayMpich+ofi [Column 7]
28 x 4 (1 node)	1403s	1438s	1417s	1437s	1644s	2014s	1728s
56 x 4 (2 nodes)	618s	669s	606s	669s	777s	984s	778s
112 x 4 (4 nodes)	367s	389s	328s	351s	445s	623s	446s
224 x 4 (8 nodes)	335s	435s	232s	337s	363s	446s	380s
392 x 4 (14 nodes)	345s	543s	212s	357s	351s	381s	395s

Execution time of OpenRadioss with different configurations

Takeaways (from our tests)

- First of all, we find that performance scaling is better with lustre file system as opposed to nfs4 file system. This is because OpenRadioss is IO intensive and thus benefits from the lustre parallel file system that provides considerable better IO bandwidth/throughput as compared to the NFS4 file system used otherwise.
- On the NFS4 file system, OpenMPI starts witnessing performance degradation beyond 4 nodes, whereas Cray-mpich continues to show some scaling until 8 nodes.
- Alongside lustre file system, we find that Cray-mpich experiences even better scaling with decent scaling upto 8 nodes whereas OpenMPI still almost stops scaling at 4 nodes with Cray-mpich outperforming OpenMPI by 1.6x and 1.7x at 8 and 14 nodes, respectively.

Session 4: **Choosing among the available compilers for an Application**

Mr. James Edwards

Weather Forecasting and MPAS-A

- Weather forecasting services is another increasingly important industry given the increasing need for weather solutions across industries.
- Weather forecasting applications are very complex mathematical and physics-based models that require a large amount of processing power and memory that has been traditionally provided by on-prem supercomputers.
  - Recently, some of the commercial weather forecasting services are increasingly migrating to the cloud to use the latest hardware on demand.
- We use the Model for Prediction Across Scales (MPAS), a cutting-edge weather model that uses a unique approach to simulate atmospheric processes at different scales.

MPAS Application

- While weather applications are usually well known for portability issues such as difficulty with compilation using different compilers, MPAS-A is much more amenable to different compilers

- We run the model at 120-km mesh resolution (40962 horizontal grid cells) for 20-day forecast with the mesoscale reference suit physics to capture the complex interactions between atmospheric processes like radiation, convection, and cloud formation

- Goal of the Lab: Maximize performance using different compilers at hand

Lab #5

- Attendees will be offered the source code and instructions to build with three different compilers – cray, intel and gnu. The builds finish within 5-7 minutes.
- All tests are performed on Milan nodes
- Attendees then run the three binaries using the provided scripts and capture performance differences
- While the first test uses –O3 optimization with all compilers, we perform another test with –Ofast optimization that additionally enables fast math optimizations for slightly reduced accuracy.
- Based on observed performance, attendees figure out the best compiler and compiler options for this application

Sample Results (from our tests)

Compilers	-O3	-Ofast
gnu	452s	-
intel	446s	424s
cray	441s	412s

Execution time of MPAS-A with different compilers and compiler options

Takeaways (from our tests)

- While all compilers perform similarly with O3, the cray compiler outperforms the intel compiler by 3% with Ofast and gnu by 10%.
  - The gnu compiler fails to run correctly with –Ofast due to loss to too much floating point precision with fast math, while this is not the case with cray and intel compilers

- The better performance with Ofast usually stems from reduced accuracy in floating point calculations, which is often acceptable especially with the accompanying performance boost.

Session 5: **How to best distribute a parallel job across nodes, sockets and cores**

Mr. Ayad Jassim

Computational Fluid Dynamics and OpenFOAM

- Computational fluid dynamics is a branch of fluid mechanics that uses numerical analysis and data structures to visualize the effect of a gas or liquid on the object it flows past.
- Used by a variety of industries such as automobile, aerospace and even sports equipment manufacturers to reduce the drag and friction from air and thus improve speed and efficiency of their products.
- We use OpenFOAM, one of the top three CFD packages world wide given its significant user base, and covers a wide range of engineering and scientific problems involving fluid flows, heat exchange and chemical reactions.

OpenFOAM Application

- While OpenFOAM is IO intensive, we find that it does scale very well with the number of ranks/nodes.
  - We thus use it to study parallel performance using different rank binding and distribution policies
- We run the OpenFOAM tutorial case of the motorBike 10M cell model that studies an incompressible steady state flow around a motorbike and rider. The model is decomposed into a number of sub-domains. Each domain is then allocated to a parallel process to solve using the OpenFOAM potentialFoam and simpleFoam solvers executable models.
- Goal of the Lab: Maximize parallel performance using combination of rank binding and distribution policies

Lab #6

- Attendees will be offered an OpenFOAM binary that could be run on upto 16 nodes of the AMD Milan processor
- Attendees then run the binary using various combinations of binding and distribution policies
- Based on observed performance, attendees figure out the best combination of binding and distribution policies for this application

Sample Results (from our tests)

Binding	Distribution	Ranks = 1 x 128	Ranks = 2 x 128	Ranks = 4 x 128	Ranks = 8 x 128	Ranks = 16 x 128
default (none)	default (block:cyclic)	607s	280s	118s	52s	28s
threads	default (block:cyclic)	1211s	528s	222s	99s	46s
cores	default (block:cyclic)	516s	216s	91s	46s	25s
sockets	default (block:cyclic)	575s	252s	111s	55s	29s
auto	default (block:cyclic)	510s	210s	85s	41s	23.4s
auto	block:block	511s	209s	85s	41s	22.7s
auto	block:cyclic	510s	210s	85s	41s	23.4s
auto	cyclic:block	510s	211s	87s	42s	25s
auto	cyclic:cyclic	509s	211s	87s	43s	25s

Execution time of OpenFOAM with different bindings and distributions

Takeaways (from our tests)

- The ‘core’ and ‘auto’ binding policies perform considerably better than others. This is because the former binds tasks to individual cores (one thread) per core and the latter binds a task to the threads (two in this case) of a core - in both cases, since the number of tasks per node are equal to the number of cores on the node, this results in optimal binding and optimal performance.

- Among the different distribution policies, we do not find the same fluctuation in performance as with binding policies. Overall, ‘block:block’ performs the best while ‘cyclic:cyclic’ performs the worst.

Session 6: **Optimizing for power and energy efficiency**

Dr. Lei Huang

Oil and Gas And Specfem3D

- In oil and gas exploration, seismic modeling is used to understand the presence, nature and size of subsurface rock layers. It involves sending seismic waves underground and used for onshore exploration.
- SPECFEM3D simulates acoustic (fluid), elastic (solid), coupled acoustic/elastic, poroelastic or seismic wave propagation in structure or unstructured conforming mesh of hexahedra.
- It can model seismic waves propagating in sedimentary basins or any other regional geological model following earthquakes. It can also be used for non-destructive testing (as opposed to blast testing) or for ocean acoustics

Specfem3D Application

- Specfem3D is both compute intensive and memory intensive
- We use it to study how to minimize power and energy
  - Memory intensive regions do not experience performance degradation even at reduced frequency or power
- We run the example case with Specfem3D which model forward simulation with 8-node mesh elements for 50000 time steps
- Goal of the Lab: Minimize power and therefore energy usage

Lab #7

- Attendees will be offered a Specfem3D binary and run instructions
- Attendees then run the binary at different clock frequencies and collect power/energy data using power monitoring counters on nodes within devCloud
- Based on observed performance, power and energy, attendees figure out the best operating frequency to minimize energy to solution

Sample results and takeaways (from our tests)

Clock	Execution time	CPU power (W)	CPU energy (kJ)	Node power (W)	Node energy (kJ)
max (2.9GHz)	112	441	49	790	87.8
2.4 GHz	131	340	45.3	666	88.6
1.9 GHz	164	277	45.3	591	96.7
1.4 GHz	209	232	47.4	539	110.1

- At the CPU/socket level, the best operating frequency from an energy point of view is 2.4GHz
  - The CPU/socket consumes nearly 8% less energy at 2.4GHz than at 2.9GHz, which is the default operating frequency.
- At the node level, however, the best point of operation is still the maximum frequency since power consumption by components (i.e. memory, etc.) other than the cores remains constant at different frequencies.

Thank You!