A Brief History of HPC Simulation and Future Challenges

Kishwar Ahmed, Jason Liu (Florida International University)
Abdel-Hameed Badawy (New Mexico State University)
Stephan Eidenbenz (Los Alamos National Laboratory)

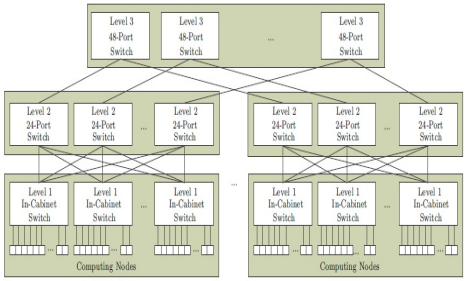
Winter Simulation Conference 2017, December 3-6, Las Vegas, NV, USA

Outline

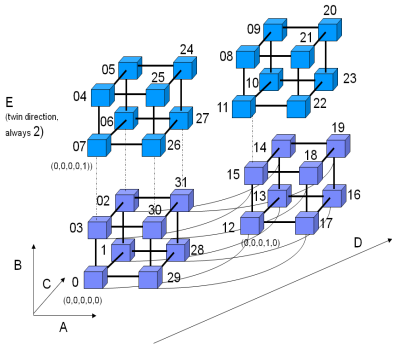
- Why HPC system simulation?
- Existing HPC system simulators
  - Processor simulator, memory simulator, interconnection simulator
  - Tools for HPC applications
- Future challenges and proposals for HPC system simulation

Why HPC Simulation?

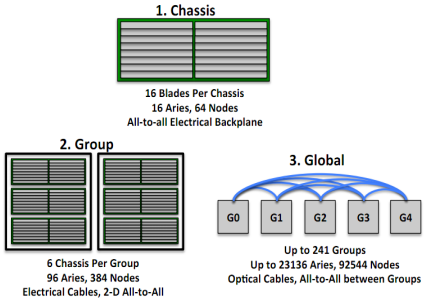
- We're rapidly approaching towards exascale computing
  - Containing thousands of nodes with high-processing capacity
  - New and advanced interconnect architecture to support high-computation capacity



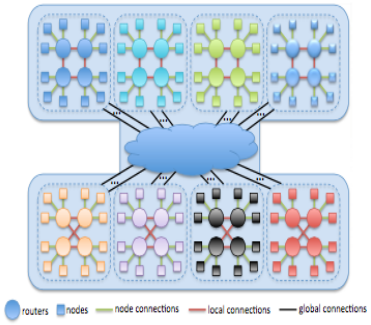
Fat-tree



Torus



Dragonfly



Slim Fly

And more…

Why HPC Simulation?

- Rapid changes in HPC architecture. For example,
  - Many-core and multi-core architecture
  - Complex memory hierarchies: uniform and non-uniform memory architecture
  - Deep pipelining, prefetching, speculative execution methods
- Performance prediction facilitates
  - Comparing (newer) design alternatives
  - Identifying performance issues of code on novel HPC platforms
  - Evaluating the whole-system impact when new components are introduced

Our Goals

- Provide a brief history of existing modeling/ simulation efforts on HPC systems
- Present unique characteristics (e.g., support for power and energy consumption) of HPC system simulators
- Outline some challenges for HPC system simulation and propose plans to overcome those challenges

Contents

- Why HPC system simulation?
- Existing HPC system simulators
  - Processor simulator, memory simulator, interconnection simulator
  - Tools for HPC applications
- Future challenges in HPC system simulation

Simulation of Processors

- Processor architecture in HPC system has gone through the most changes
  - Introduction of many-core and multi-core architecture
  - Support for various instruction sets
  - Arrival of accelerator technologies (e.g., GPUs)
- Many processor simulators exist
  - How many instructions can be executed per second? (scalability)
  - How many cores they can support? (scalability)
  - How accurately they can replicate instruction execution? (accuracy)

Simulation of Processors (Contd.)

- RSIM (1997)
  - Only multicore processor available at the time
- SimpleScalar (2002)
  - Supported almost all the complex interactions (e.g., complex branch prediction schemes)
  - Various instruction set architectures (ISAs) (e.g., Alpha ISA)
- gem5 (2011)
  - Simulate multicore system with varying degree of accuracy and speed
  - Accommodates many sub-components (on-chip interconnection, GPGPUs)
  - Main advantage:
    - A community research project, that is highly-extensible
    - Supports various ISAs (e.g., Alpha, SPARC, x86, ARM)

Simulation of Processors (Contd.)

Simulator Name (year)	What it does?	Accuracy	Scalability	Highlights	Remarks
McSimA+ (2013)	Uni-core, multi-core-many-core simulator	*Good accuracy* when compared with published results and real machine runs	Scalable to processor with *thousands* of cores	Lightweight, detailed, flexible cycle-accurate simulator	(+) good accuracy and scalability (+) supports simulation of **heterogeneous** architecture
Zsim (2013)	Large-scale many-core simulator	Accurate through leveraging instruction-driven timing models and leveraging dynamic binary translation	Fast and scalable, through running in parallel; can simulate *1024-core* chip	Fast, accurate and scalable many-core simulator	(+) **large-scale** simulation capability

Simulation of Processors (Contd.)

Simulator Name (year)	What it does?	Accuracy	Scalability	Highlights	Remarks
Manifold (2014)	A parallel multi-core simulator	*No comparison* with existing models	Up to *64 core* simulation	Component-based design; power, thermal and energy models	(+) easy extensibility (+) core-level **power and energy** consumption (-) Accuracy not tested (-) Scalability not shown too good

Simulation of Memory

- Memory is also going through rapid changes
  - Increase in memory capacity
  - Different technologies, such as DRAM to non-volatile memory
- There exist many memory simulators
  - Compare with other memory simulators? (scalability or speedup and accuracy)
- Early efforts on memory simulation
  - The Wisconsin Wind Tunnel (1993)
    - A stepping stone for cache-based memory simulation
  - CACTI (1996)
    - Capable of memory model hierarchy simulation at various levels: registers, buffers, caches

Simulation of Memory (Contd.)

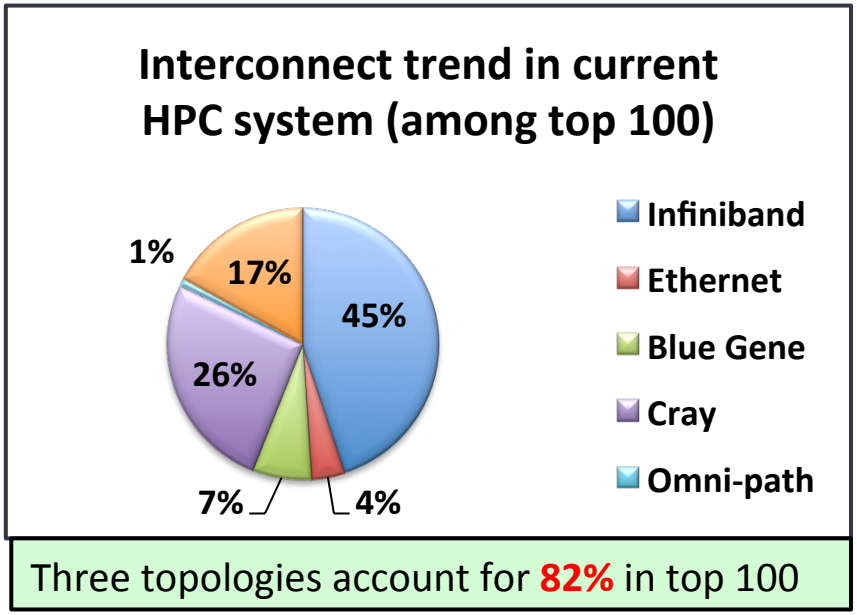
Simulator Name (year)	What it does?	Accuracy	Scalability	Interoperability	Highlights	Remarks
DRAMSim2 (2011)	Simulate DDR II and DDR III memory systems	Compared with micron verilog output: no discrepancies	Compared to MARSx86, 30% simulation time increase	Straightforward integration with MARSx86	- easy-to-integrate and accurate - simple programming interface and object oriented design	(+) good accuracy (+) easy-to-integrate (-) high simulation time to achieve high accuracy
Ramulator (2015)	DRAM simulation, but with focus on easy-extensibility	Validated using Verilog model: no violations were reported	2.5 times faster than next fastest simulator (USIMM)	Two versions: 1) standalone 2) integrated with gem5	-**extensible**: support for various existing and future simulators -modular design	The simulator is both **fast and accurate** compared to the existing memory simulators.

Simulation of Memory (Contd.)

Simulator Name (year)	What it does?	Accuracy	Scalability	Interoperability	Highlights	Remarks
NVMain (2012)	Simulation of both DRAM main memory and **non-volatile memory**	Compared with DRAMSim	--	With CACTI and NVSIM to estimate power etc.	- models **endurance** of a non-volatile memory - more **flexibility** (e.g., compared to DRAMSim)	+ Both DRAM and non-volatile memory simulation. + Ideal for prediction of power consumption of different memory systems.

Simulation of Interconnects

- Dominant interconnection network topologies in current HPC systems: **Fat-tree**, **Torus, and Dragonfly**
- Compare different interconnect simulators
  - Scalability: How many ranks or cores can simulate?
  - Accuracy: How close are the results compared to previous results?



Simulation of Interconnects

Simulator Name (year)	What it does?	Accuracy	Scalability	Highlights	Remarks
BigSim (2004)	PDES-based large-scale simulator	Simulation time and execution time within 6% range, during actual running of Jacobi 3D on Blue Gene/L	64,000 simulated processors	PDES-based MPI, AMPI (Advanced MPI) simulator	(+) A **mature and well-established** PDES-based simulator (-) Limited congestion-handling capability
Structural Simulation Toolkit (2011)	PDES-based large-scale simulator	Focused on validation from October 2014	--	An all-inclusive simulation framework (i.e., memory, interconnect, CPU)	(+) Ideal for system simulation with large-scale interconnect. (+) Can be used for cases when **energy-prediction** is a requirement.

Simulation of Interconnection Network (Contd.)

Simulator Name (year)	What it does?	Accuracy	Scalability	Interoperability	Remarks
Extreme-scale Simulator (xSim) (2010)	PDES-based simulator with various MPI function implementations	Close latency resemblance for a small experimental setup: pingpong technique	1.048 million ranks (**MPI hello world program**)	--	(+) large-scale accurate interconnect simulation (-) runs simple programs to demonstrate scalability
Co-Design of Exascale Storage System (CODES) (2011)	ROSS-based simulator for hardware and software models of HPC systems	The accuracy has shown to be perfect in most comparison	1 billion ranks, 16,384 cores	Have been run with an existing storage system prototype	(+) support for various interconnection types with **various level of fidelity**

Simulation of Interconnection Network (Contd.)

Simulator Name (year)	What it does?	Accuracy	Scalability	Interoperability	Remarks
FatTreeSim (2015)	A CODES-based Fat-tree interconnect simulator	less than 10% error rate when compared to Emulab (pingpong benchmark)	305 million events/s	Running with YARNsim	The simulator is ideal for **large-scale fat-tree** interconnect simulation of both HPC and data center system.
Performance Prediction Toolkit (PPT) (2015)	PDES-based simulator, including various interconnection network	Validated against empirical studies and actual machine runs	Simulated 156,672 MPI ranks	Various models (such as, GPU, memory models)	(+) extensive validation (+) Fully integrated with all standard MPI calls

Modeling HPC Applications

- Vampir
  - A performance analysis tool for parallel MPI/OpenMP applications
  - Support program instrumentation
    - Different types of programs (sequential, MPI, OpenMP, hybrid MPI and OpenMP)
    - Various types of instrumentations (compiler, library, runtime, manual)
- Tuning and Analysis Utilities (TAU)
  - A (well-established, flexible, portable, robust) performance instrumentation, measurement, analysis, and *visualization* framework
  - *Flexible* instrumentation capability
    - Allows users to select performance instrumentation at levels of application code

Modeling HPC Applications (Contd.)

- HPCTOOLKIT
  - Application performance measurement, analysis, and presentation toolkit for both sequential and parallel applications
  - Measurement ability for a number of derived performance metrics
    - E.g., peak and actual performance rather than raw data
- Analytical models
  - PALM
    - Analytical performance model for parallel applications
    - Performs static and dynamic analysis of the source code
  - ASPEN
    - A *domain-specific* language for analytical performance modeling
    - Formal definition includes
      - Application behavior (e.g., parameters, kernels, control flow)
      - Abstract machine (e.g., node, interconnect, cache, memory, core)

Contents

- Why HPC system simulation?
- Existing HPC system simulators
  - Processor simulator, memory simulator, interconnection simulator, simulation of HPC applications
- Future challenges in HPC system simulation

Future Challenges

- We're in the "wild-west" stage of development!
  - A few *individual* blocks for hardware, middleware, and software building blocks
  - Many of them are not compatible with each other
  - Some are open-source, but many are *closed-source*
- Most of the simulation models appear *after* novel architecture has been introduced
  - No opportunity to perform *early*, *cost-efficient* assessment of novel ideas

Five-Step Plan

- Establish clearly-defined use cases
- Agree on a single tool
- Build and maintain comprehensive model library of all hardware and software components
- Ensure reproducibility
- Extend to newer HPC architecture

Establish Clearly-Defined Use Cases

- Early assessment of **hardware technologies** and concepts
  - E.g., new caching strategies or speculative execution methods
- Early assessment of **algorithmic variations** for middleware software and application software. For example,
  - Basic functionality of task-based parallelism runtimes (such as Legion or HPX)
  - Algorithmic variations of large computational physics code

Establish Clearly-Defined Use Cases (Contd.)

- Apply simulation modeling during procurement of the **new** HPC system
  - Currently, relies heavily on the *expert opinions* of both buyers and sellers
  - Modeling will help to remove any kind of biases
- **Bottleneck** resource identification through sensitivity analysis across parameters. For example,
  - In the hardware side: increasing or decreasing cache sizes for instances
- There'll always be a trade-off between model scalability and accuracy
  - Use cases need to find a well-established balance in this trade-off space

Agree on a Single Tool

- In most successful simulation community, there's an agreement on a *dominant* tool and then build on that as a community effort. For example,
  - Communication network simulation: NS-2 (or NS-3)
- We could feel necessity of three different community amalgam for HPC system simulation
  - Application and middleware software tool
  - Interconnect model
  - Compute node models
- A single tool should emerge as a result of such efforts

Build and Maintain a Comprehensive Model Library

- *Development mode:* We should focus on building a comprehensive *easy-to-use* library
  - Allowing non-expert users to quickly build composed model of hardware and software components to test
- *Maintenance mode:* Once there is a large user base with stable library version
  - Quickly model and assess emerging technologies
- Architecture community already operates in this fashion
- Credibility of models need to be established
  - Run validations whenever possible

Ensure Reproducibility

- We need to ensure that results are reproducible
  - E.g., existence of standard input formats
- A *detailed description* of reproducibility of results
- Use different tools to produce the same results
  - If such results hold, credibility increases

Going Beyond Traditional HPC Architecture

- We should not just get constrained within simulation of traditional HPC architecture
- Should aim for novel HPC architectures and model for performance gains even before they're available
  - Quantum computing
  - Neuromorphic computing
  - Inexact computing

Conclusions

- We presented briefly efforts on HPC system simulation at various system and sub-system level
- We outlined some of the future challenges in HPC system simulation
- We presented some plans to tackle these challenges

A Brief History of HPC Simulation and Future Challenges

Kishwar Ahmed, Jason Liu (Florida International University)
Abdel-Hameed Badawy (New Mexico State University)
Stephan Eidenbenz (Los Alamos National Laboratory)

Thank you!