Towards Exascale Computing: An Automatic Hardware-Software Co-design Framework for Aspen

Mariam Umar+, Jeremy S. Meredith (Advisor)*, Jeffrey S. Vetter (Advisor)*, Kirk W. Cameron(Advisor)+
Department of Computer Science, Virginia Tech+, *Oak Ridge National Laboratory, TN
{mariam.umar,kirk.w.cameron} @vt.edu,{jsmeredith,vetter}@ornl.gov

Motivation

- HW-Software co-design is a challenge as we move towards Exascale era
  - Existing solutions e.g., DVFS, memory throttling are becoming less efficient
- Problem is exacerbated when:
  - Implementing application specific hardware
  - Integrating hardware with software
  - Changing application/hardware specifications at runtime

Our Approach:

- Automated hardware software co-design for Aspen, which generates
  - Automated application model for Aspen
  - Automated Machine Model for Aspen
- Allows us to
  - Develop and test application without requiring real hardware for future and exascale systems
  - Obtain portable solution, which can be applied to a range of current and future architectures
  - Avoid programmer's involvement, minimal overhead
  - Assist in approximate computing

Aspen and Our Hardware Software Co-design Framework

- **Aspen** is a domain specific language (DSL) that uses special grammatical semantics to represent models by generating:
  1. **Application Model**: used to measure flops, loads, stores etc.
  2. **Machine Model**: finds theoretical peak performances etc.
- Aspen is fast, portable and accurate [3]
- Hardware Software Co-design framework
  - Automatic Application Model generator
  - Automatic Machine Model Generator

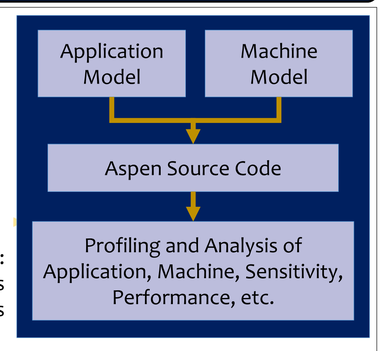


Figure 1: Aspen's components

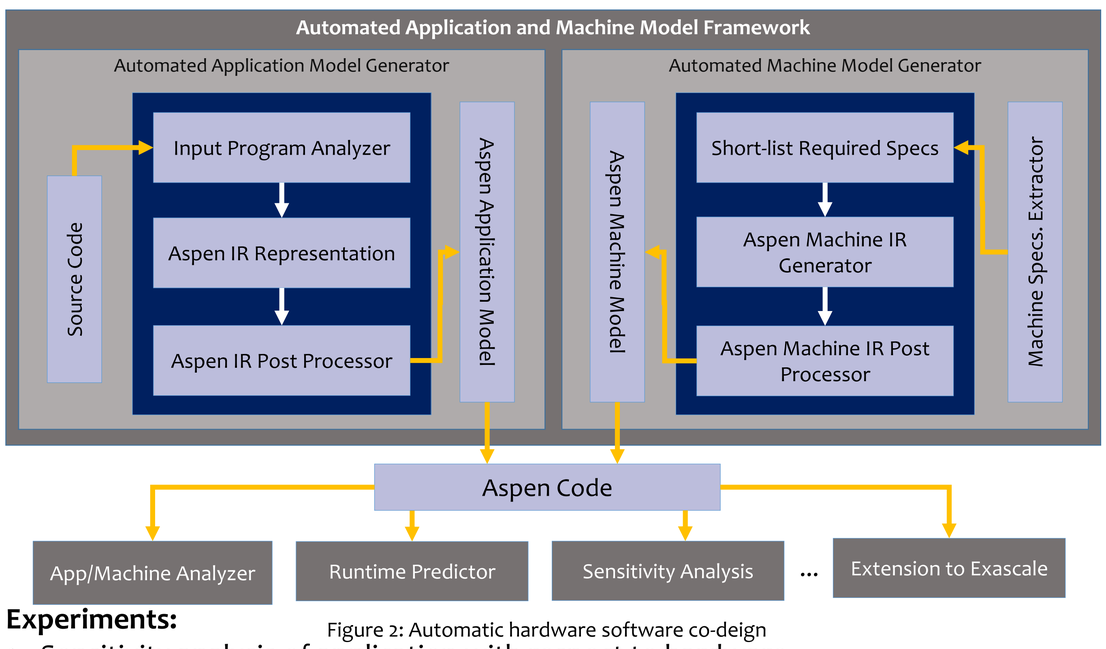


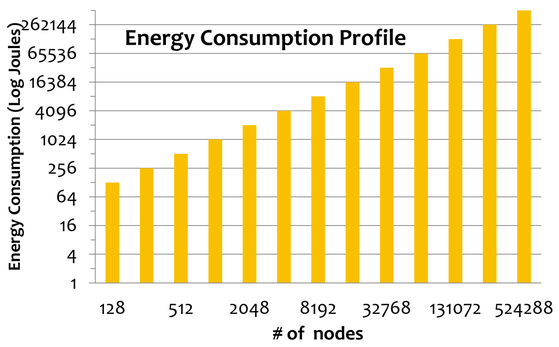
Figure 2: Automatic hardware software co-deign

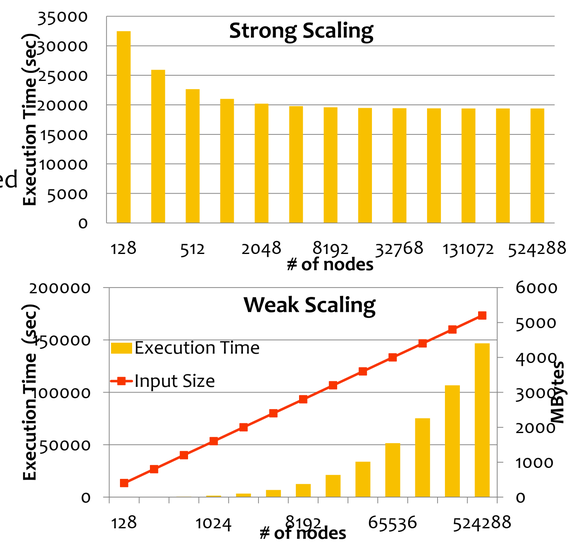
Experiments:

1. Sensitivity analysis of application with respect to hardware
2. Strong and weak scaling and energy profiling of LULESH on Titan
3. Strong and weak scaling and energy profiling of LULESH on Future Exascale systems

Strong & Weak Scaling with Energy Profiling for Titan

- Application : LULESH
  - Is amenable to strong scaling
  - Execution time decreases until 4096 nodes
- Energy Profile
  - Linear increase with increasing nodes
  - Energy budget dictates # of nodes to be used

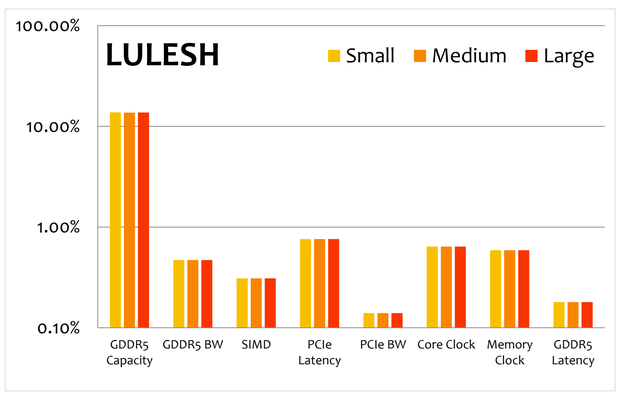


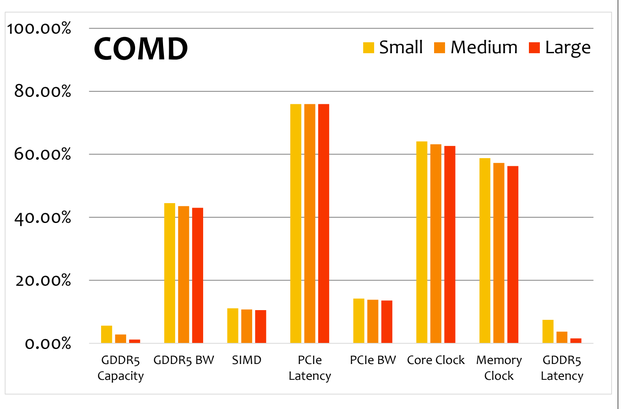


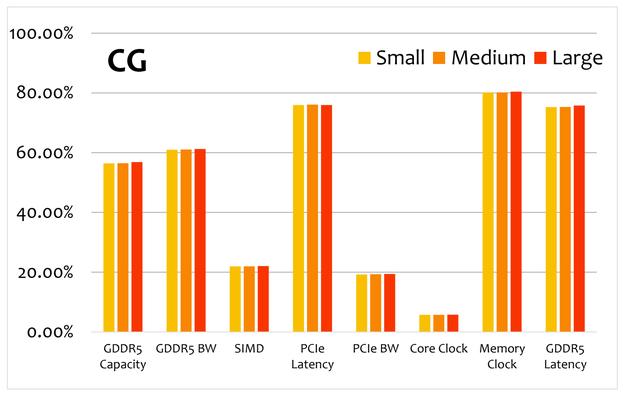
Sensitivity Analysis

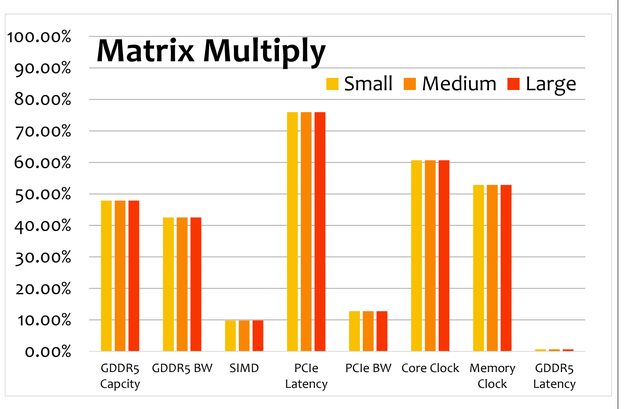
CPU	Core Clock	Memory	Memory BW	GPU	GPU Memory	GPU Clock
Intel I7-6660	2.4 GHz	32 GB	34.1 GB/sec	Tesla K20c	5GB	2.6 GHz

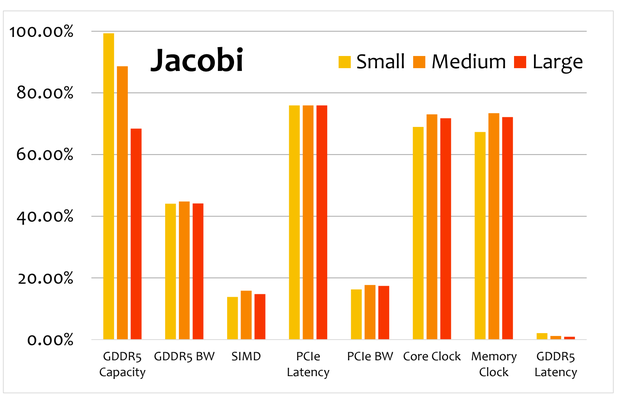
- Baseline is vendor-specifications of underlying hardware
- Y-axis for sensitivity analysis results is % difference with respect to baseline
- Small, medium and large input sizes are inspired by SHOC benchmarks, e.g., Matrix Multiply, small=1K, medium=500K, large=1M

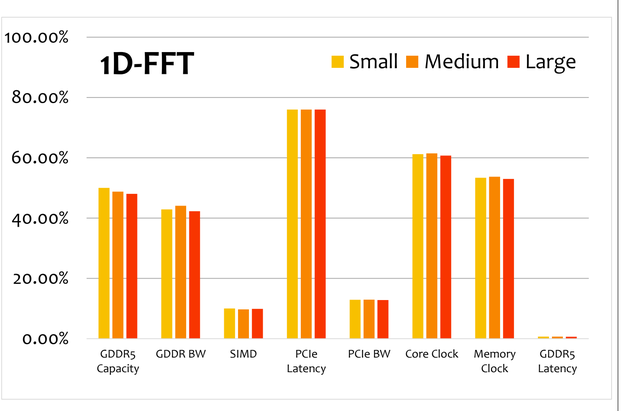












What-If Analysis

Application	Dominant hardware parameter?	Why?	Improvements?
Lulesh	Not significantly impacted by change in hardware	Tendency to not change because of different properties incurred by ~40 kernels	Application specific improvements e.g., loop fusion, array contraction etc.
CoMD	PCIe Latency, Core Clock, Memory Clock	Memory and communication intensive	DVFS, memory throttling, Application specific optimizations etc.
CG	Memory clock, GDDR5 Latency, PCIe latency	Memory and communication intensive	Global memory latency, improving global memory bandwidth, application specific optimizations etc.
Matrix Multiply	PCIe latency, Core clock, Memory Clock	Memory and communication intensive	DVFS, memory throttling, application specific optimization e.g., loop unrolling etc.
Jacobi	GDDR5 capacity, PCIe latency, core clock, memory clock	Memory-Intensive nature and much communication involved	DVFS, memory throttling, application specific optimizations e.g., optimizing use of on-chip memory etc.
1D-FFT	PCIe Latency, Core Clock, Memory Clock	Memory-Intensive nature and much communication involved	DVFS, memory throttling, application specific optimizations e.g., use of on chip memory, etc.

Strong & Weak Scaling with Energy Profiling for Future Exascale Systems

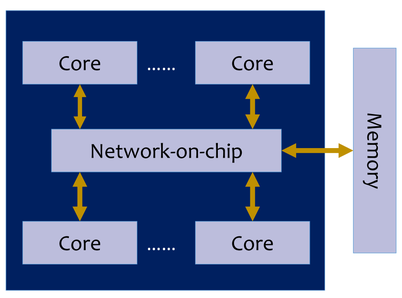
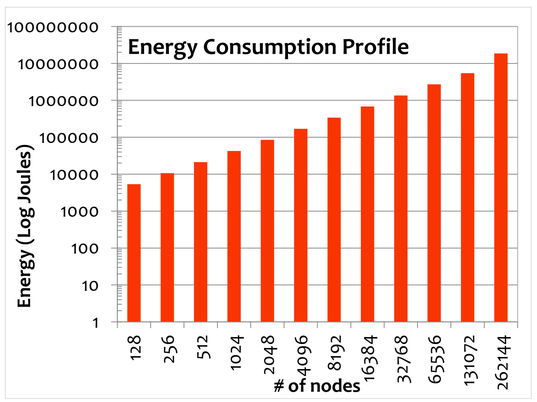
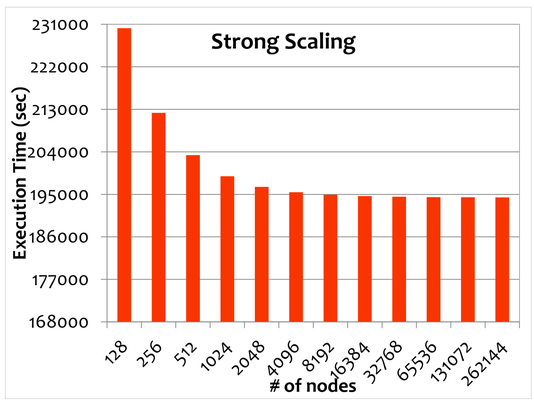
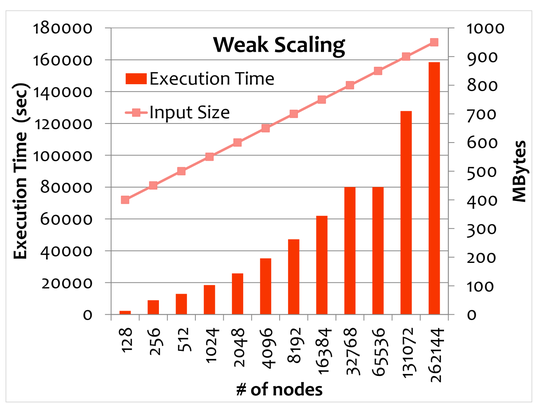


Figure3: Abstract Machine Model for Future Exascale system [2]







- Application: LULESH
- Strong Scaling: Execution time increases with increasing cores, and do not stabilize until 131K
- Weak Scaling: Execution time shows linear trend with increasing nodes and increasing input size
- Energy profile: Energy consumption show linear trend but stays less than 20MW as required by DOE

Acknowledgements

Authors would like to thanks Seyong Lee (ORNL) regarding his guidance for Compass. This material is based upon work supported in part by the NSF Grants No. 1422788, 0910784 and 0905187. The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DE-AC05-00OR22725. This research was supported in part by an ASTRO appointment at the Department of Energy's Oak Ridge National Laboratory, through the Oak Ridge Institute for Science and Education.

References

[1] S. Lee, J. S. Meredith, and J. S. Vetter, "COMPASS: A framework for automated performance modeling and prediction," ICS-2015.
[2] Ang, J. A., et.al., Abstract Machine Models and Proxy Architectures for Exascale Computing. Co-HPC '14, New Orleans, Louisiana.
[3] K. L. Spafford and J. S. Vetter, "Aspen: a domain specific language for performance modeling," SC12.