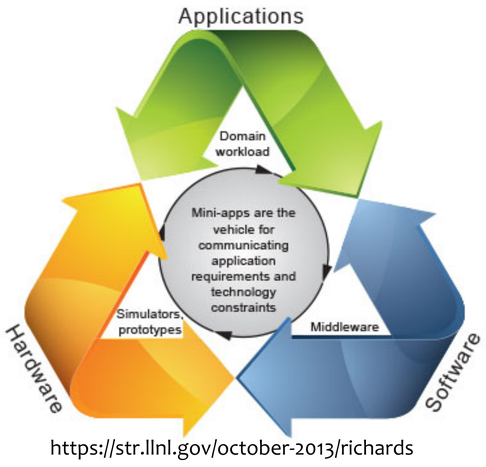
An Automatic Hardware-Software Co-design Framework for Sensitivity Analysis of Proxy Applications

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

Motivation

- HW-Software co-design is a challenge as we move towards more complex architectures and applications for exascale era:
  - Existing solutions e.g., DVFS, memory throttling are no more efficient
- Problem is exacerbated when:
  - Implementing application specific hardware
  - Integrating hardware with software
  - Changing application/hardware specifications at runtime



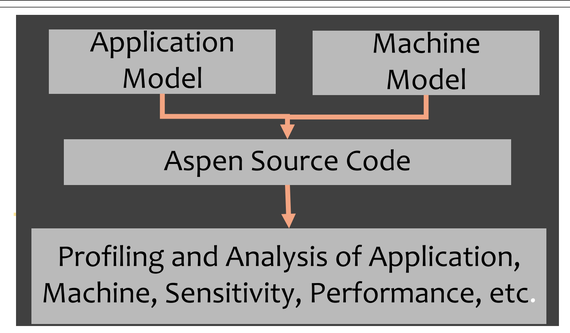
https://str.llnl.gov/october-2013/richards

Our Approach:

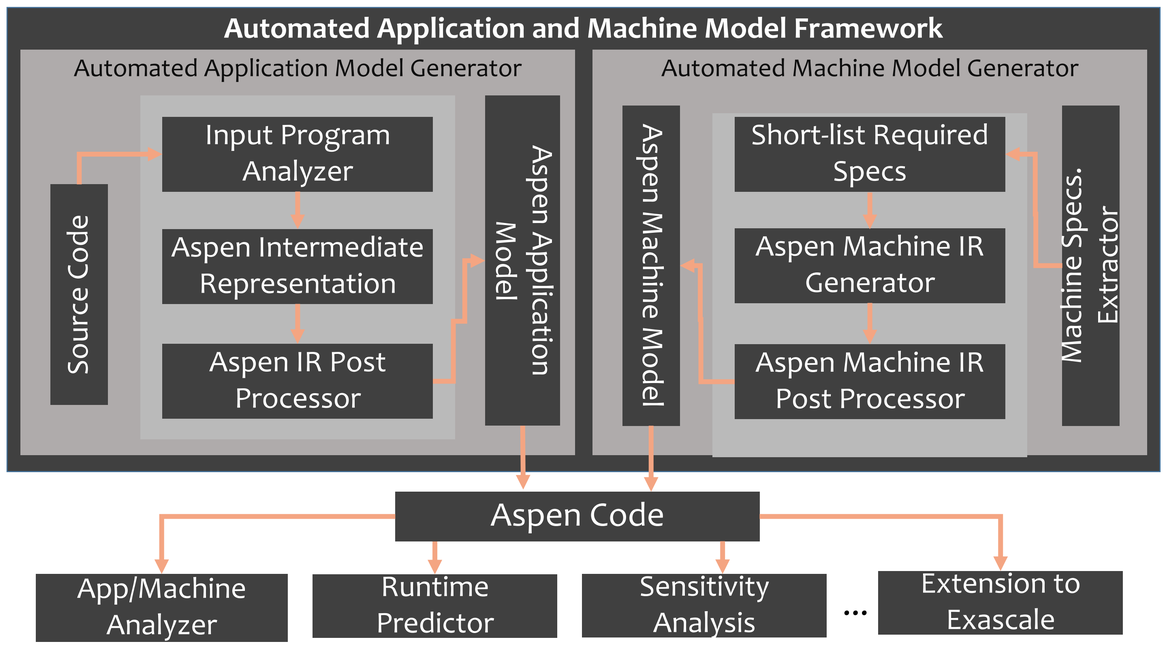
- Automated hardware software co-design for Aspen, which generates
  - Automated application model for Aspen
  - Automated Machine Model for Aspen
- Allows us to
  - Develop and test application without requiring real hardware for future and exascale systems
  - Obtain portable solution, which can be 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

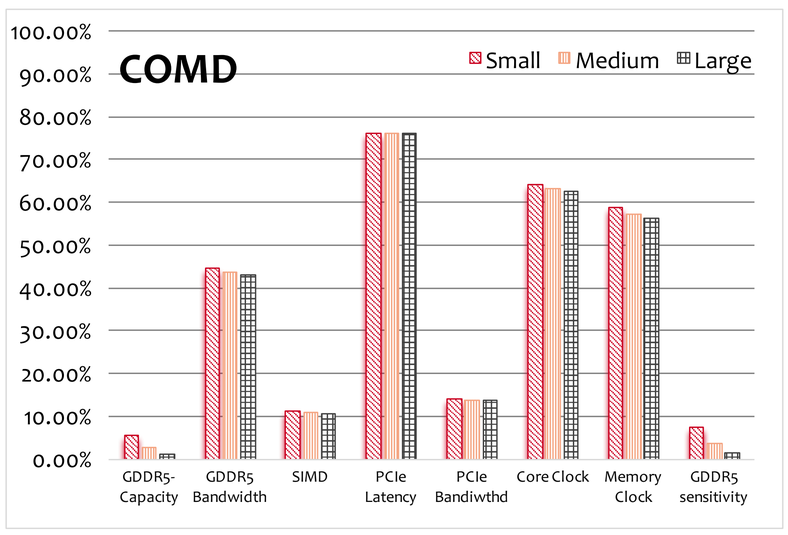
- Aspen is a domain specific language (DSL) that uses special grammatical semantics to represent the performance models by generating:
  1. **Application Model**:
     - used to measure flops, loads, stores etc.
  2. **Machine Model**:
     - finds theoretical peak performances etc.
- Aspen is fast,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

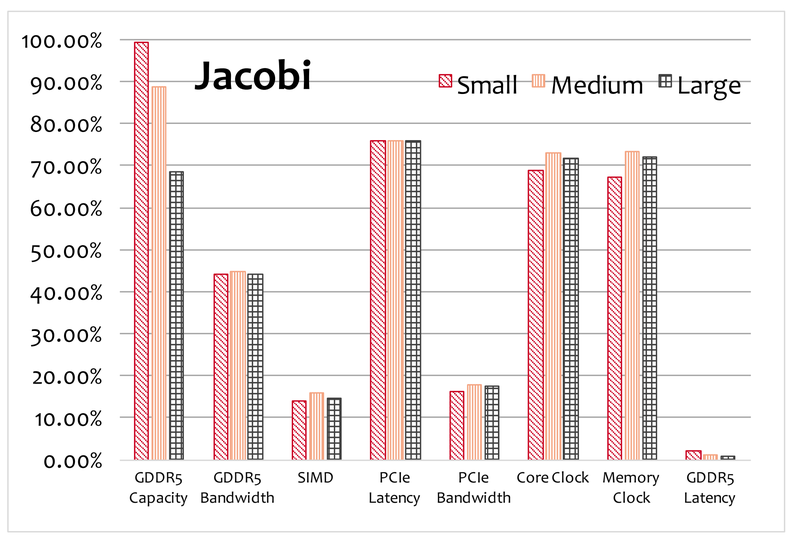


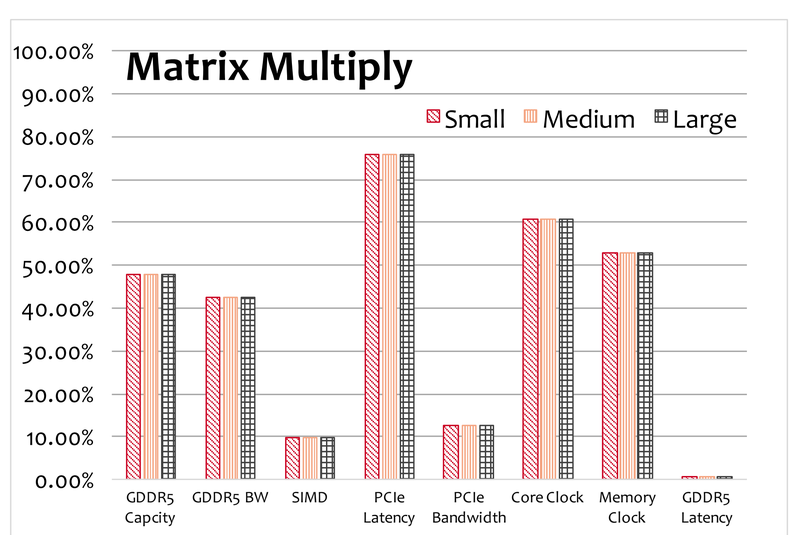
- Hardware Software Co-design framework consists of two components:
  - Automatic Application Model generator
    - Analyze source code
    - Extract application characteristics e.g., FLOPS, loads, stores etc.
    - Generate Aspen's intermediate representation(IR)[1]
    - Process Aspen's IR to produce Aspen's post processed IR
  - Automatic Machine Model Generator
    - Extract machine specifications using Linux user and kernel functions
    - Generate Aspen's intermediate representation
    - Insert Aspen's grammar and process it to produce post processed IR



Results and Analysis







What-If Analysis

Application	Dominant hardware parameter?	Why?	Improvements?
CoMD	PCIe Latency, Core Clock, Memory Clock	Memory and communication intensive	DVFS, memory throttling, 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.

Acknowledgements

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.