Uniform Abstractions for Heterogeneous Parallel Systems

Vikram Adve

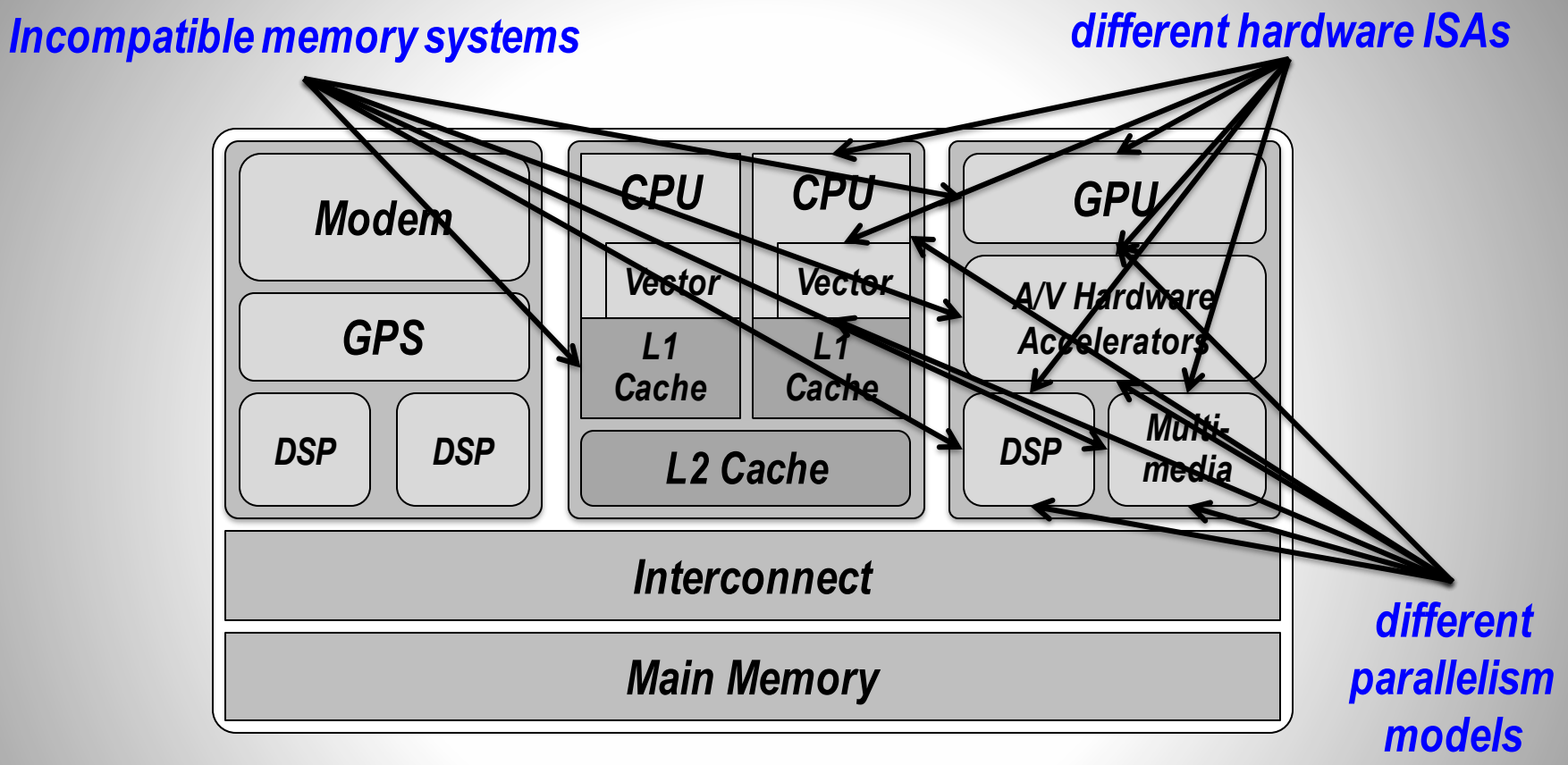
With:

Maria Kotsifakou, Prakalp Srivastava, Adel Ejjeh, Hashim Sharif, Matt Sinclair, Rakesh Komuravelli, Sarita Adve and Sasa Misailovic

University of Illinois at Urbana-Champaign

Supported by: NSF, SRC, DARPA, Intel

A Modern Mobile SOC



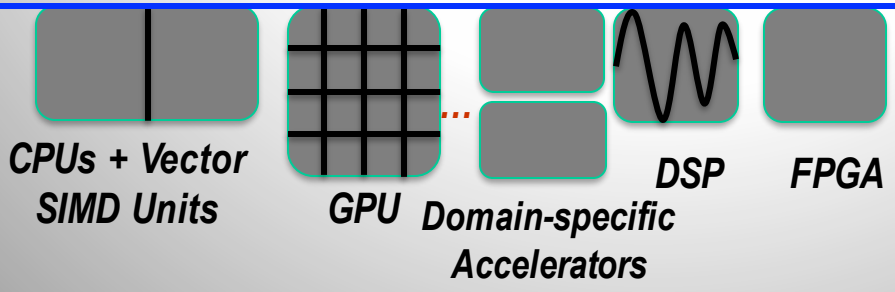
And different SoCs have *different combinations* of such hardware!

Key to Programmability:

Common abstractions for heterogeneous parallel hardware

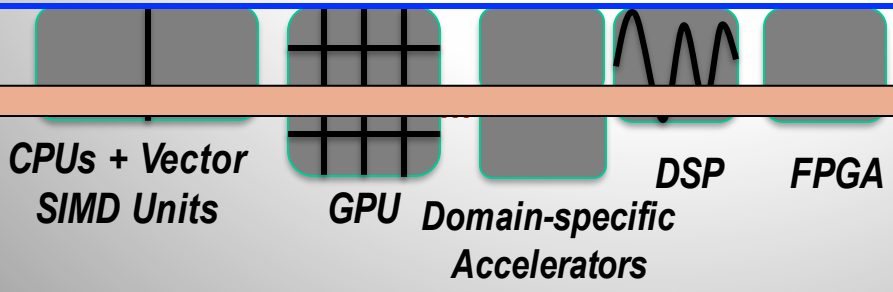
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia*
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



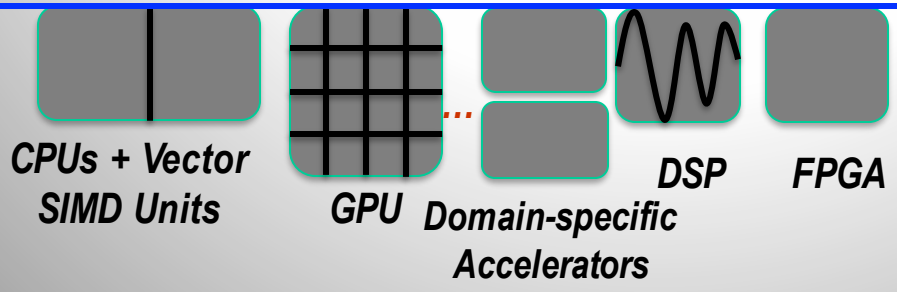
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia*
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	***"Hardware" ISA***	***GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …***



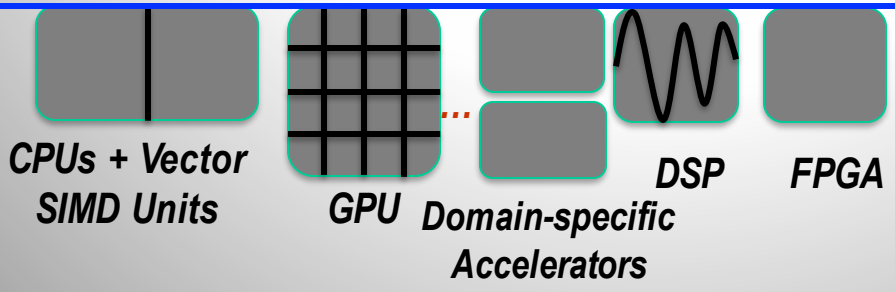
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia*
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	***Virtual ISA***	***IBM AS/400, PTX, SPIR-V HSAIL, HPVM***
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



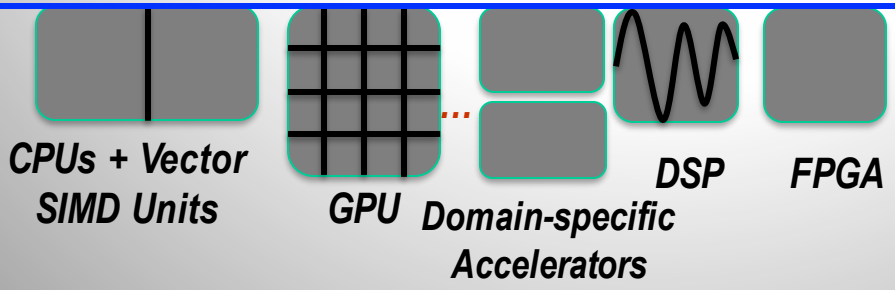
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia*
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	***Language-neutral Compiler IR***	***Delite IR, HPVM, MLIR***
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



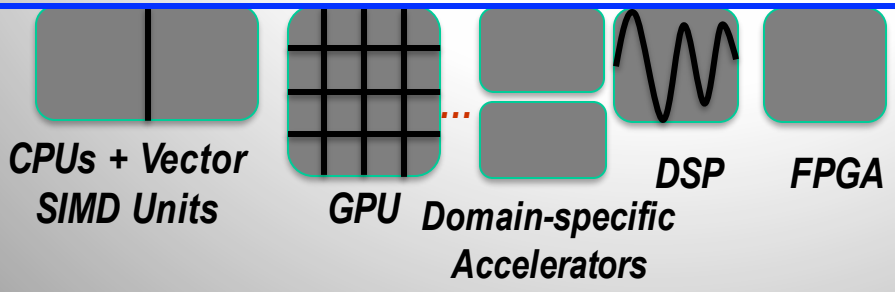
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc,*
Language innovation	***Language-level Compiler IR***	***Delite DSL IR, DLVM, TVM, …***
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



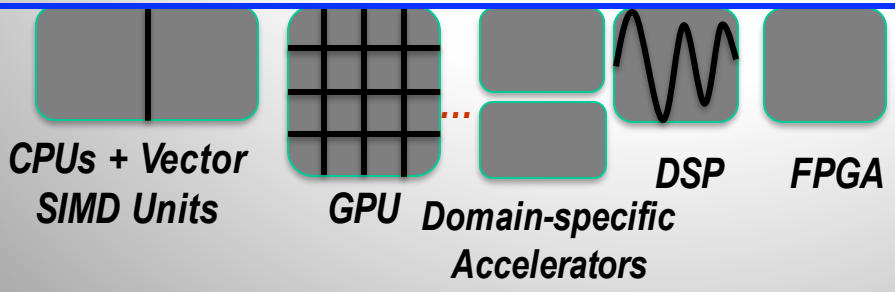
Interface Levels and Key Benefit

App. productivity	*Domain-specific prog. language*	*TensorFlow, MXNet, Halide, …*
App. performance	***General-purpose prog. language***	***CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia***
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



Interface Levels and Key Benefit

App. productivity	***Domain-specific prog. language***	***TensorFlow, MXNet, Halide, …***
App. performance	*General-purpose prog. language*	*CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia*
Language innovation	*Language-level Compiler IR*	*Delite DSL IR, DLVM, TVM, …*
Compiler investment	*Language-neutral Compiler IR*	*Delite IR, HPVM, MLIR*
Object-code portability	*Virtual ISA*	*IBM AS/400, PTX, SPIR-V HSAIL, HPVM*
Hardware innovation	*"Hardware" ISA*	*GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, …*



Which Interface Levels Can Be Uniform?

TensorFlow, MXNet, Halide, … — **Domain-specific prog. language**

CUDA, OpenCL, OpenAcc, OpenMP, Python, Julia, … — **General-purpose prog. language**

Delite DSL IR, XLA IR, TVM, … — **Language-level Compiler IR**

Too diverse to define a uniform interface

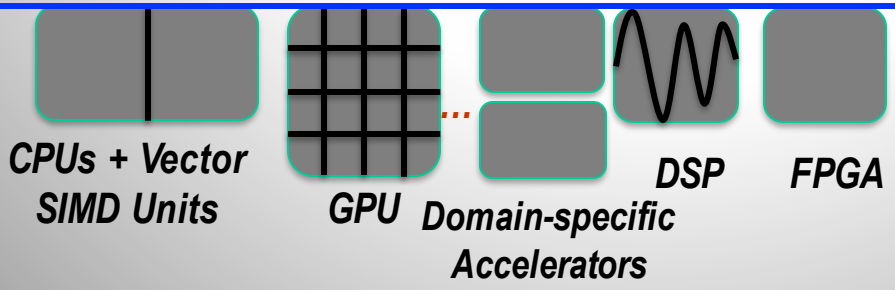
Delite IR, HPVM, MLIR — **Language-neutral Compiler IR**

IBM AS 400, PTX, SPIR HSAIL, HPVM — **Virtual ISA**

Much more uniform

GPU ISAs, SIMD ISAs, TPU, Domain-specific accelerators, … — **"Hardware" ISA**

Also too diverse …



What Should the Interface Enable?

- Uniform parallel abstraction for diverse hardware
- Aggressive compiler optimizations
- Vendor-provided back ends
- Use of target-specific low-level libraries: MKL, cuDNN, …
- Partitioning, static scheduling, dynamic scheduling
- Application-guided error vs. energy vs performance tradeoffs

WITHIN REACH: 2-5 Yrs

- H/w-agnostic HLS for FPGAs, ASICs
- Application-driven software + hardware specialization
- Mechanized formal verification of designs

10 Yr. GOALS

The HPVM Program Representation

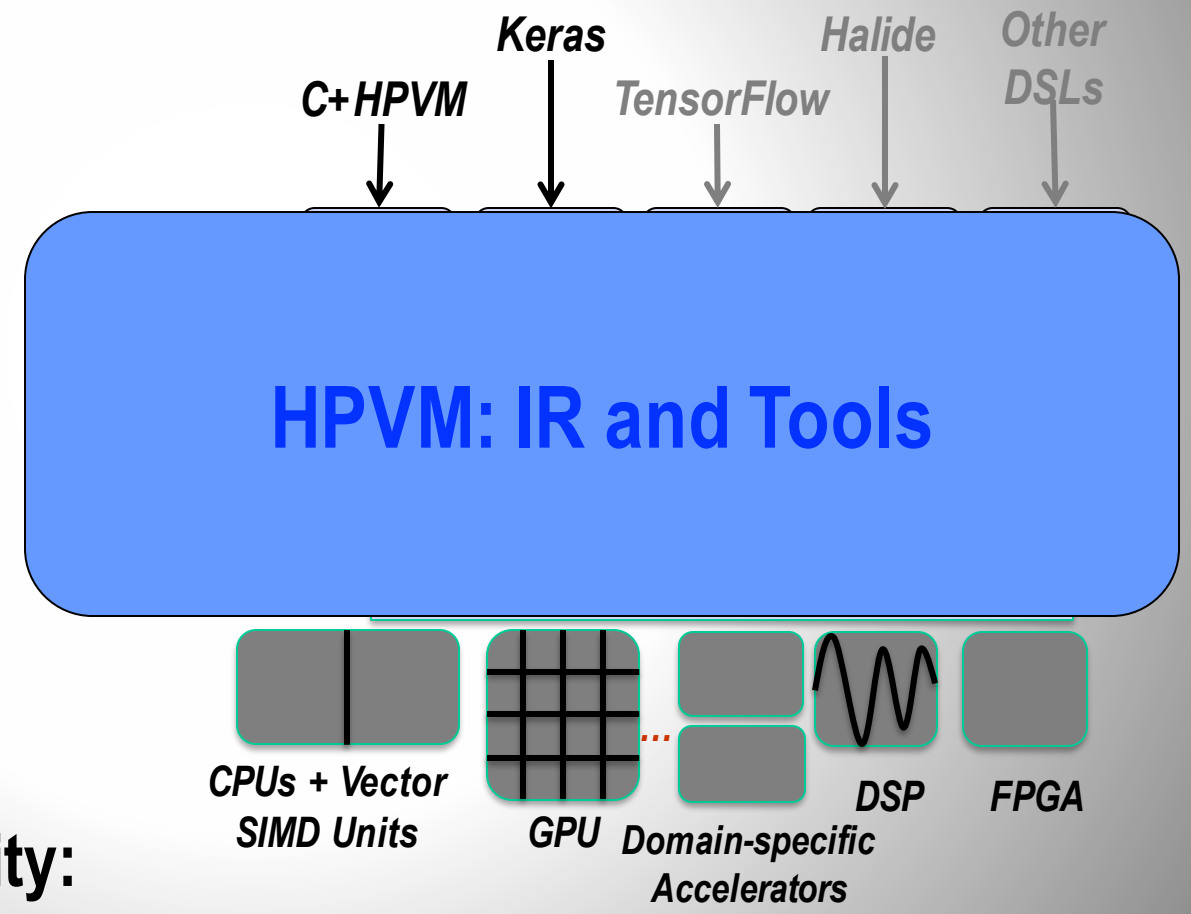
- A common parallel abstraction
- Compiler IR + Virtual ISA + Run-time scheduling

Kotsifakou et al., PPOPP 2018

Heterogeneous Parallel Virtual Machine

Goal: Programmability for Heterogeneous Parallel Systems

Mobile phone SoCs
Supercomputers
Cloud with accelerators



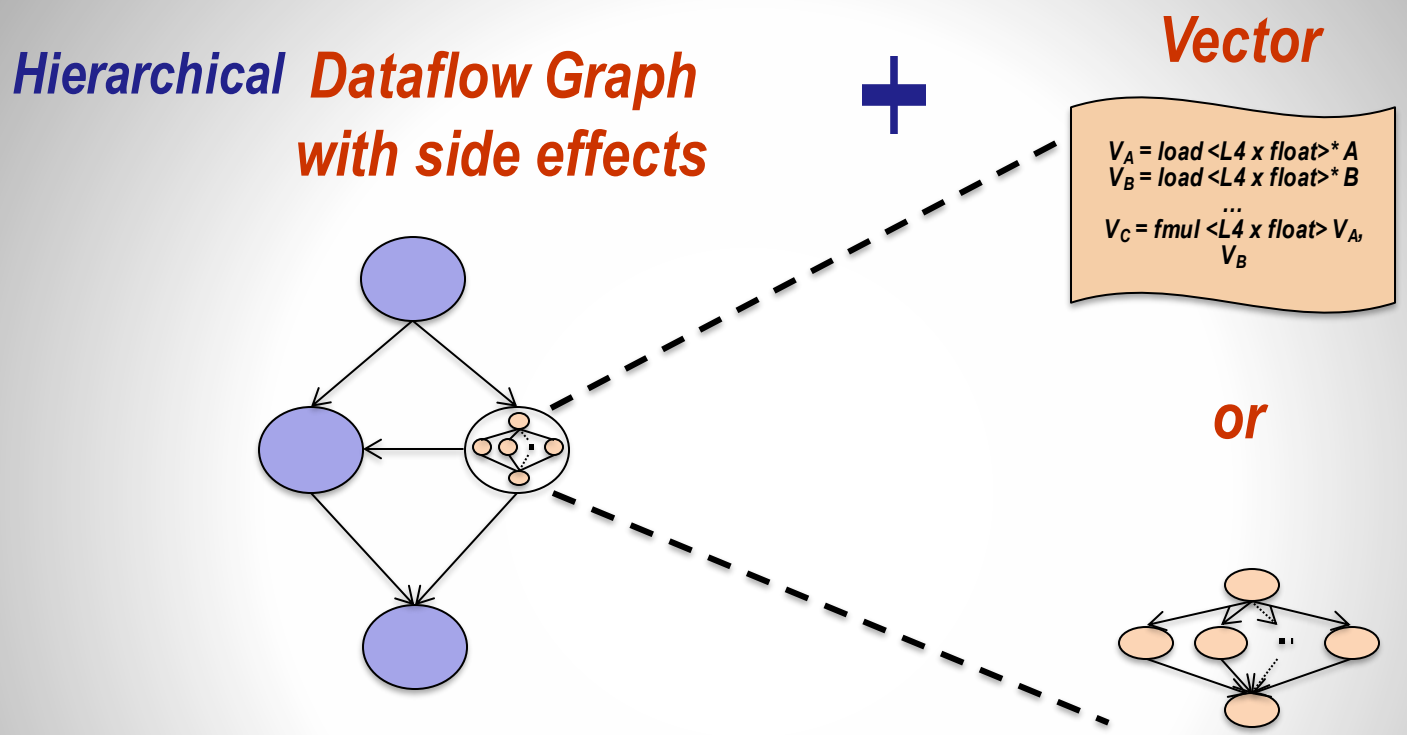
Use HPVM for:

1. Portable *object* code
2. Retargetable parallel compiler IR and system
3. Run-time scheduling

Key to Programmability:

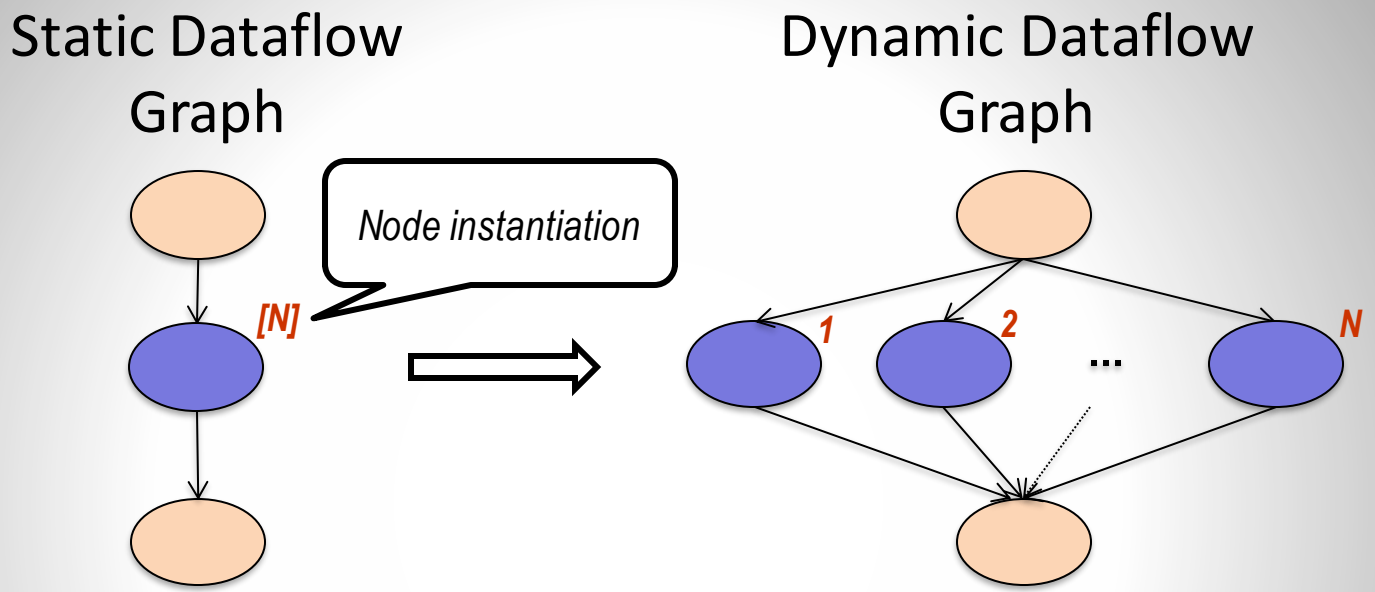
Common abstractions for heterogeneous parallel hardware

HPVM Abstraction of Parallel Computation



- Graph nodes – coarse-grain or fine-grain computational tasks
- Graph edges – explicit data transfer between nodes
- Loads and stores – implicit communication via shared memory
- Hierarchical – multiple levels of nested parallelism

HPVM Abstractions

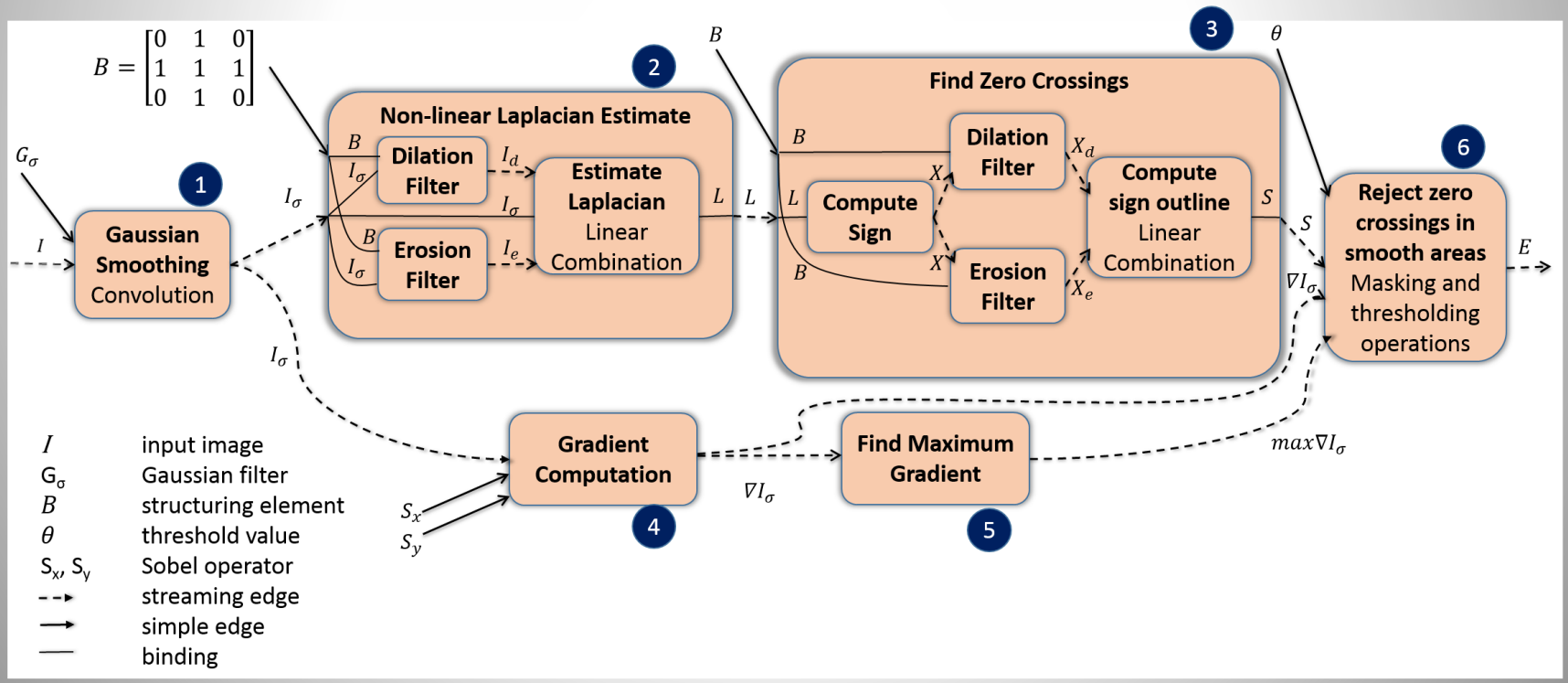


- ✓ **Graph Structure – coarse grain task parallelism, streams, pipelines**
- ✓ **Graph hierarchy – nested parallelism**
- ✓ **Node Instantiation – captures SPMD-style data parallelism**
- ✓ **Vector instructions in leaf nodes – fine grain vector parallelism**
- ✓ **Supports high-level optimizations**
- ✓ **Captures FPGAs, some semi-custom hardware**

N different parallelism models ⇨ single unified parallelism model!

E.g., Edge Detection in Images

- ✓ **Pipelined (task) parallelism with streaming input images**
- ✓ **Medium-grain data parallelism within pipeline stages**
- ✓ **Fine-grain data parallelism in most stages**

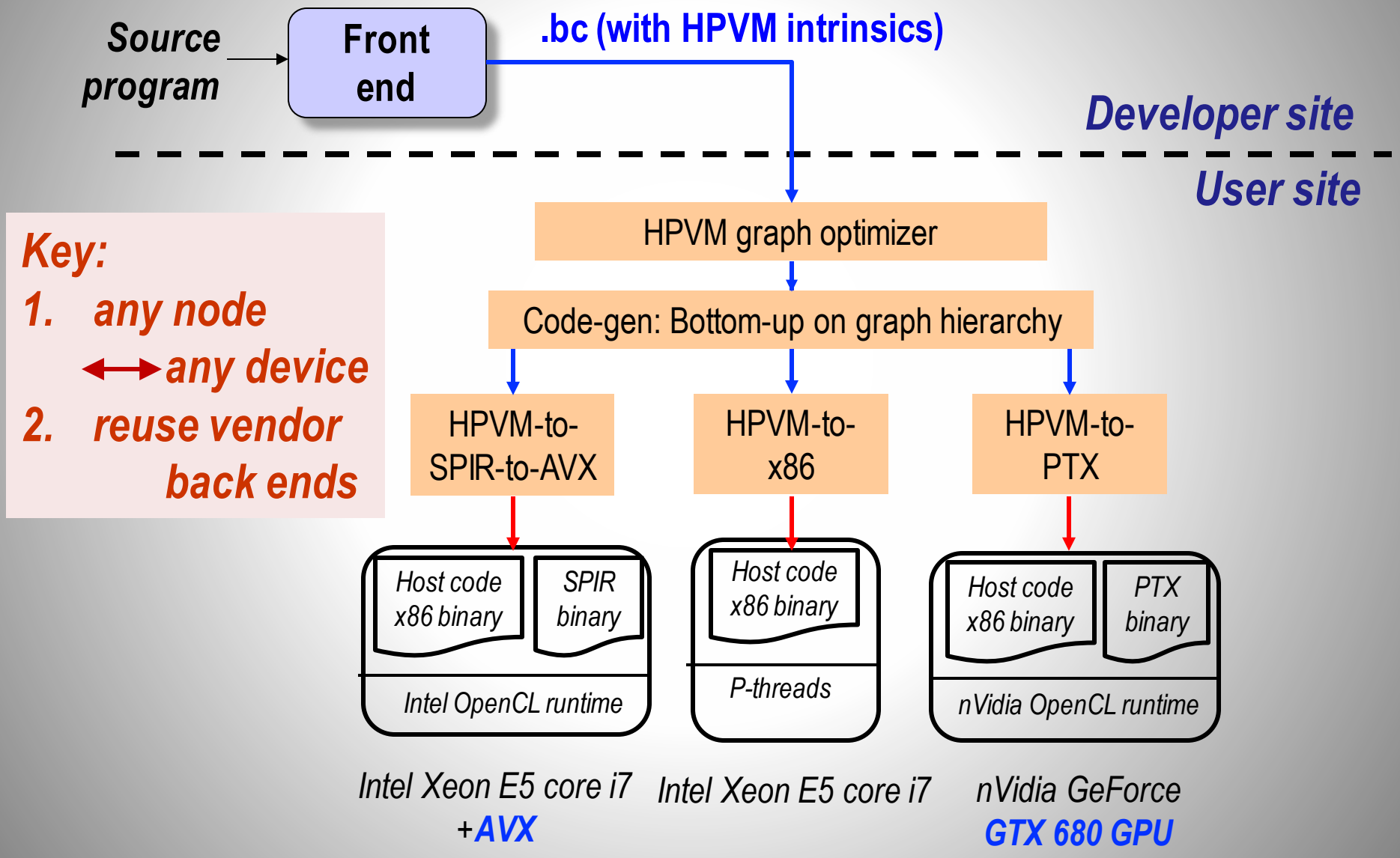


HPVM Compiler Optimizations

Complex optimizations as simple graph transforms

- Graph node “tiling” for memory hierarchy
- Graph node merging
- Graph pipelining
- Graph partitioning and mapping
- (Future) Graph-based loop optimizations

Code Generation Strategy – Overview



Evaluation: Summary

Abstraction and object-code portability

- Single HPVM code is close to (or slightly worse than) separately hand-tuned code on both GPU, AVX
- HPVM performance limited by vendor-specific back ends, *not* by HPVM abstractions

Flexible scheduling

- HPVM enables highly flexible mappings to diverse h/w

Ongoing Research (1)

Sharif et al., OOPSLA 2019

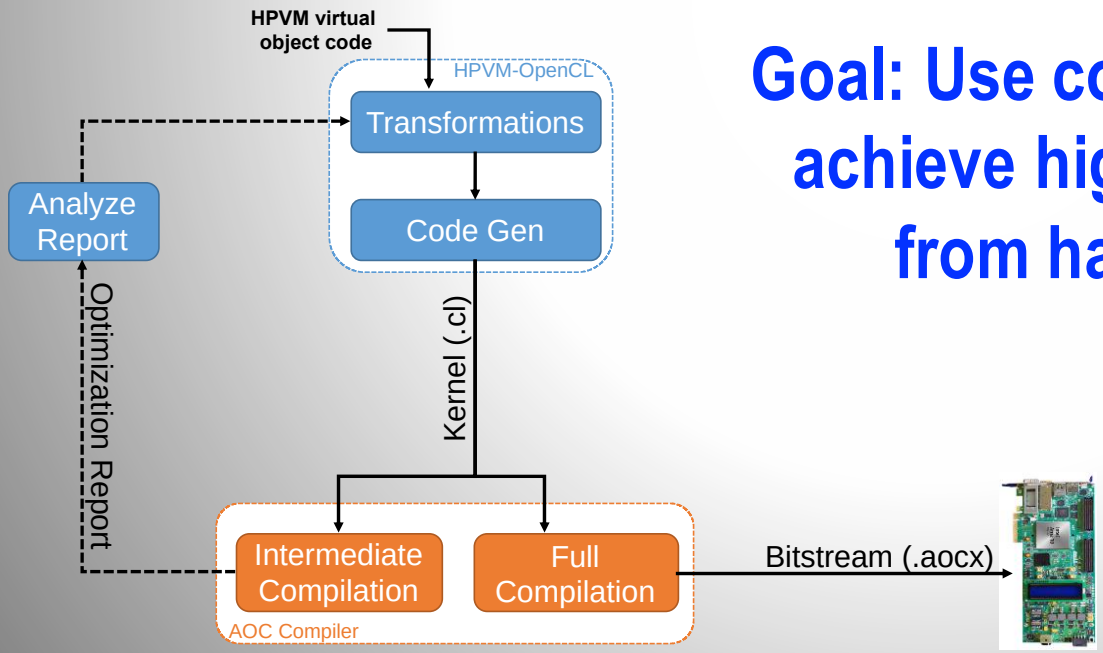
ApproxHPVM for accuracy-aware optimization

- **App developers only express end-to-end accuracy goals**
- **Domain-specific strategy:**
  - Extend HPVM with tensor domain ops
  - Express hardware-independent accuracy metrics in IR
- **Algorithmic approximations as well as system-level**
- **Portable virtual ISA after hardware-agnostic autotuning**
- **Dynamic optimization to adapt to run-time conditions**

Ongoing Research (2)

Hardware-agnostic programming of FPGAs

- **FPGAs are becoming widely available in data centers**
- **Application users lack hardware expertise**



Goal: Use compiler optimizations to achieve high-perf. FPGA designs from hardware-agnostic code

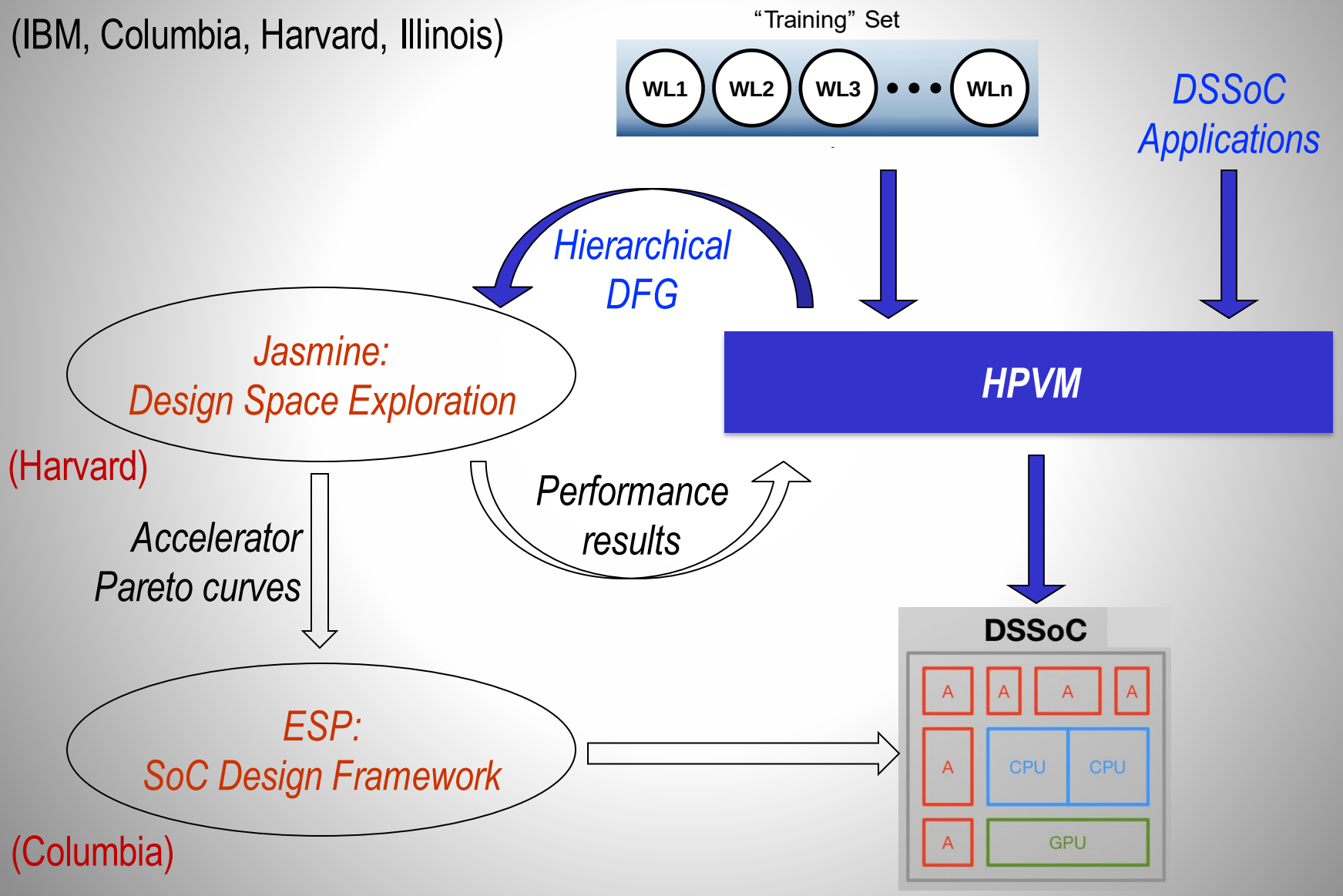
Ongoing Research (3)

Integrate ApproxHPVM with Jasmine Toolflow

- Improve hw-agnostic tuning to match hw-specific
- Partition application + iterate through design space
- Explore approximate hw, sw mechanisms

DSSoC: Hardware Design Space Exploration

(IBM, Columbia, Harvard, Illinois)



Ongoing Research (4)

Domain-specific programming of edge systems

- **Xilinx Zynq, NVIDIA Jetson Nano, Intel Movidius …**
- **Example: ARM (+ GPU) (+ FPGA) (+ DNN)**
- **Users: Crop scientists, civil engrs, medical researchers…**
- **Can we enable *non-expert users* to program complex heterogeneous SoCs?**
  - Very high-level DSLs
  - Automatic partitioning, approximation, mapping, code generation
  - Automatic run-time scheduling, performance analysis

Summary

HPVM: portability + performance for heterogeneous systems

ApproxHPVM: easy access to approximation techniques

Long-term goals:

- Application-driven **hardware design** ← needs uniform interfaces!
- Rich compiler infrastructure for **DSLs**
- **Easy programming** of energy-efficient edge compute systems

Questions?