High-Performance Domain-Specific Languages using Delite

Kunle Olukotun, Kevin Brown, Hassan Chafi, Zach DeVito, Sungpack Hong, Arvind Sujeeth

Pervasive Parallelism Laboratory
Stanford University

Tutorial Overview

- Motivation for tutorial
  - Lots of interest in DSLs
  - New ideas: DSLs for productivity and parallelism
  - New software paradigm: DSL infrastructure
- Goals
  - Introduction to performance oriented DSL development
  - DSL examples and uses
  - DSL implementation basics
  - Delite: DSL infrastructure for DSL compiler development
  - Intro to Scala: basis for Delite, and important new programming lang.

2020 Vision for Parallelism

- Make parallelism accessible to all programmers
- Parallelism is not for the average programmer
  - Too difficult to find parallelism, to debug, maintain and get good performance for the masses
  - Need a solution for "Joe/Jane the programmer"
- Can't expose average programmers to parallelism
  - But auto parallelizatoin doesn't work

Three Faces of Computing

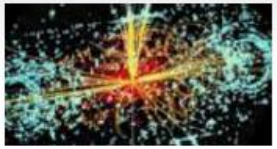
- Predicting the future
  - Modeling and simulation (weather, materials, products)
  - Decide what to build and experiment or instead of build and experiment ⇒ third pilar of science
- Coping with the present (real time)
  - Embedded systems control (cars, planes, communication)
  - Virtual worlds (second life, facebook)
  - Electronic trading (airline reservation, stock market)
  - Robotics (manufacturing, cars, household)
- Understanding the past
  - Big data set analysis (commerce, web, census, simulation)
  - Discover trends and develop insight

Explosion of Data Sources

PERVASIVE PARALLELISM LABORATORY

Experiments

Simulations



Archives

Literature

Cornell University Library

arXiv.org

Consumer

The Challenge
Enable Discovery

Deliver the capability to mine, search and analyze this data in near real time

Petabytes
Doubling & Doubling

The Response

Discovery itself is evolving

Computing Goals: The 4 Ps

- Power efficiency
- Performance
- Productivity
- Portability

Era of Power Limited Computing

- Mobile
  - Battery operated
  - Passively cooled
- Data center
  - Energy costs
  - Infrastructure costs

Power and Performance

Energy efficiency **Performance**

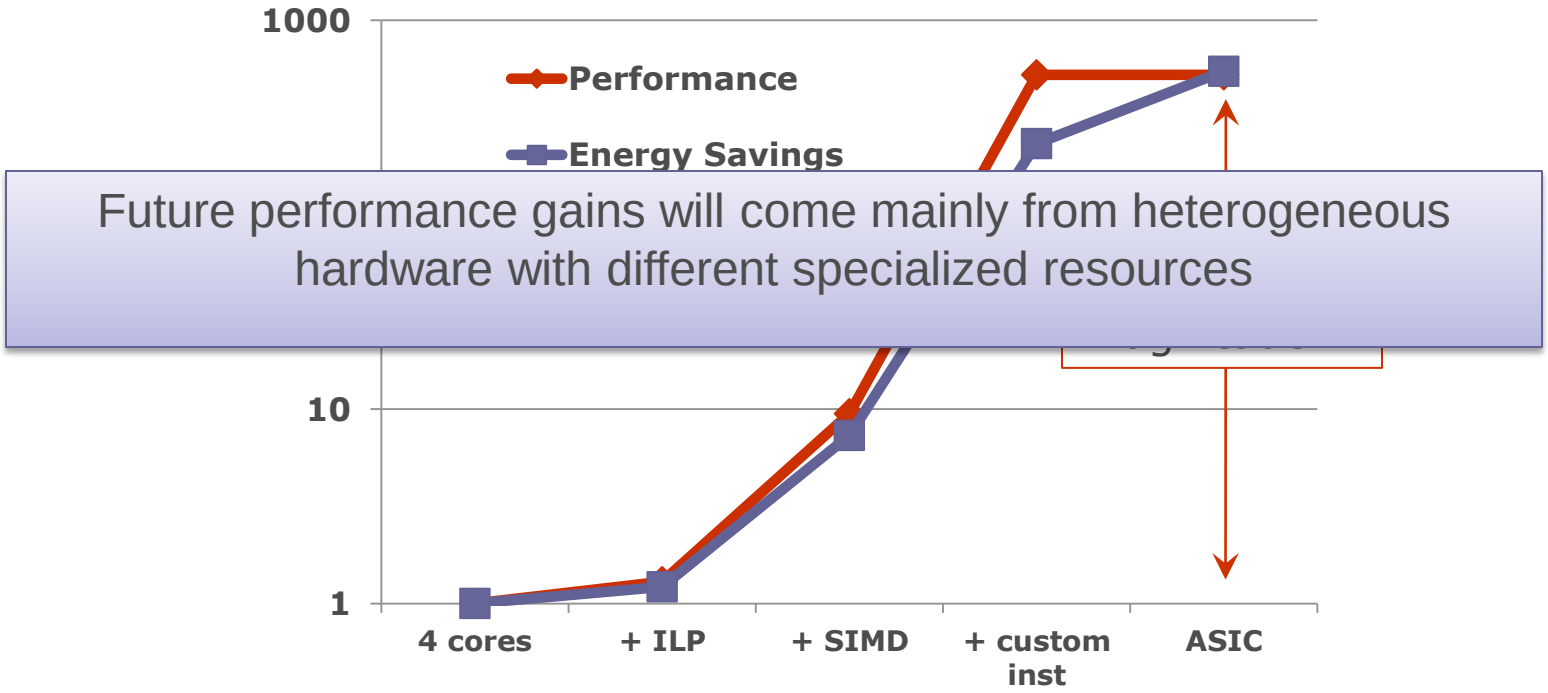
$$Power = \frac{Joules}{Op} \cdot \frac{Ops}{second}$$

FIXED

Specialized (Heterogeneous) Hardware

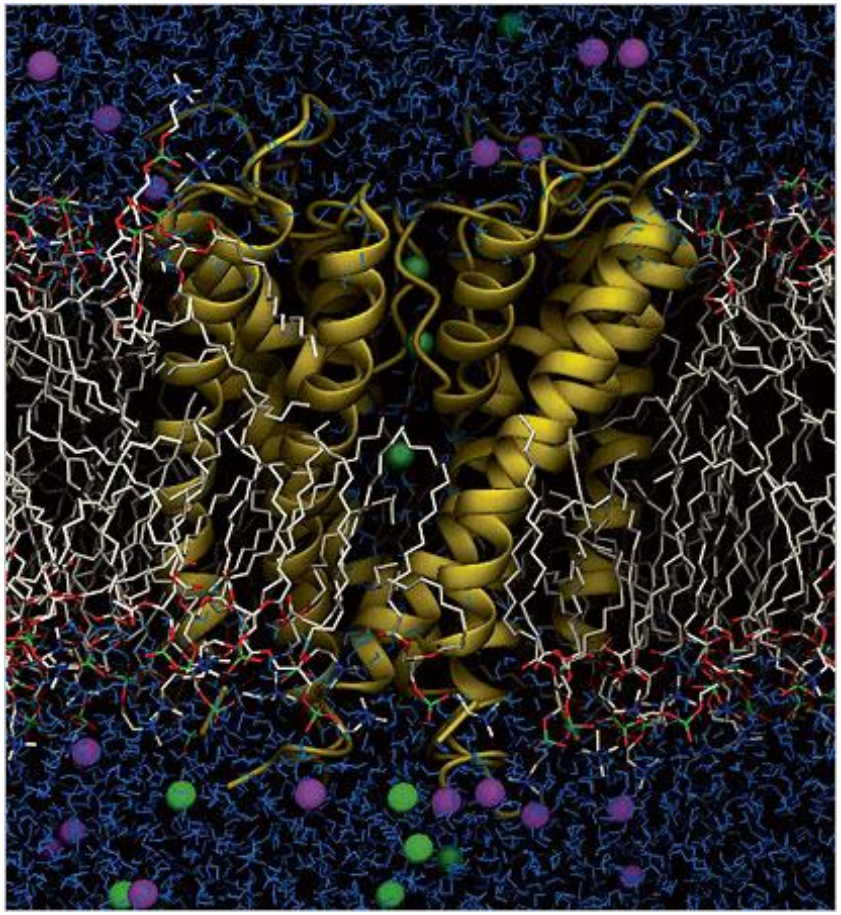
- Heterogeneous HW for energy efficiency
  - Multi-core, ILP, threads, data-parallel engines, custom engines

- H.264 encode study



Source: Understanding Sources of Inefficiency in General-Purpose Chips (ISCA'10)

DE Shaw Research: Anton

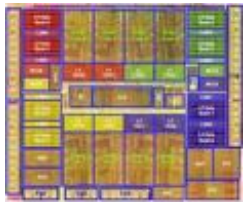


Molecular dynamics computer

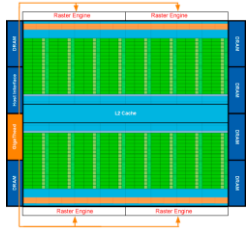
100 times more power efficient

D. E. Shaw et al. SC 2009, Best Paper and Gordon Bell Prize

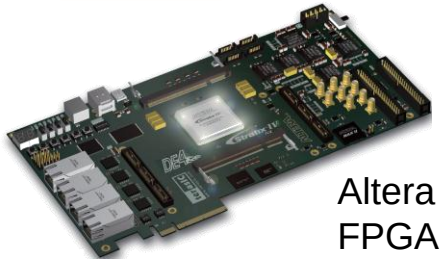
Heterogeneous Parallel Architectures Today



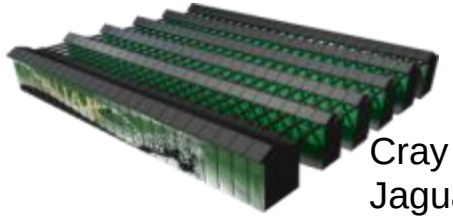
Sun T2



Nvidia Fermi



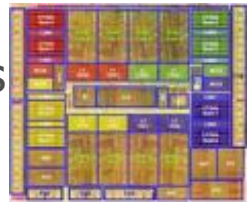
Altera FPGA



Cray Jaguar

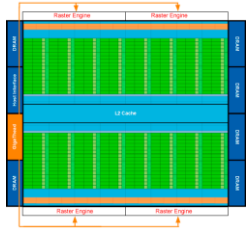
Heterogeneous Parallel Programming

Pthreads
OpenMP



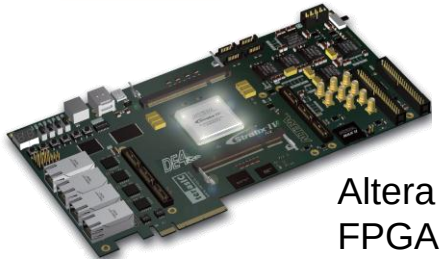
Sun
T2

CUDA
OpenCL



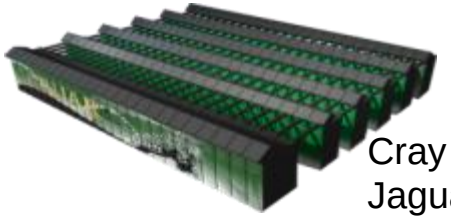
Nvidia
Fermi

Verilog
VHDL



Altera
FPGA

MPI
PGAS



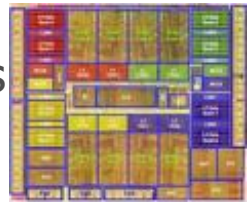
Cray
Jaguar

Programmability Chasm

Applications

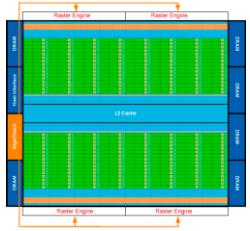
- Scientific Engineering
- Virtual Worlds
- Personal Robotics
- Data informatics

Pthreads
OpenMP



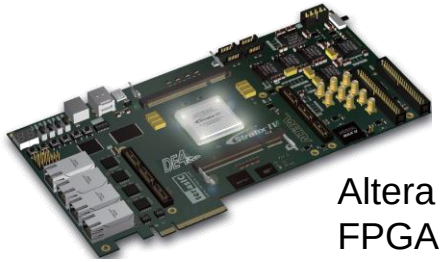
Sun T2

CUDA
OpenCL



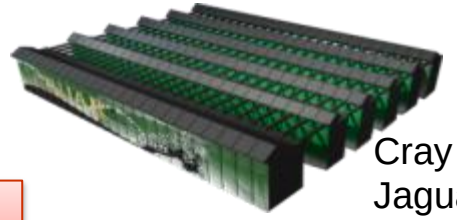
Nvidia Fermi

Verilog
VHDL



Altera FPGA

MPI
PGAS



Cray Jaguar

Too many different programming models

Hypothesis

It is possible to write one program
and
run it on all these machines

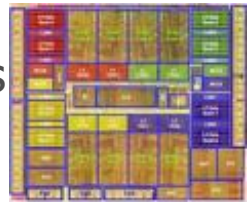
Programmability Chasm

Applications

- Scientific Engineering
- Virtual Worlds
- Personal Robotics
- Data informatics

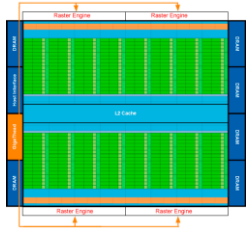
Ideal Parallel Programming Language

Pthreads
OpenMP



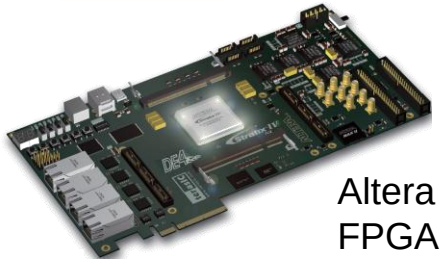
Sun T2

CUDA
OpenCL



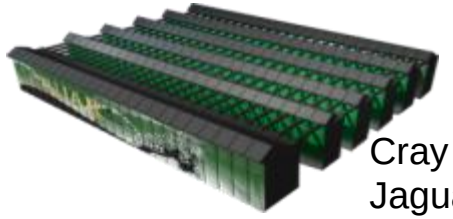
Nvidia Fermi

Verilog
VHDL



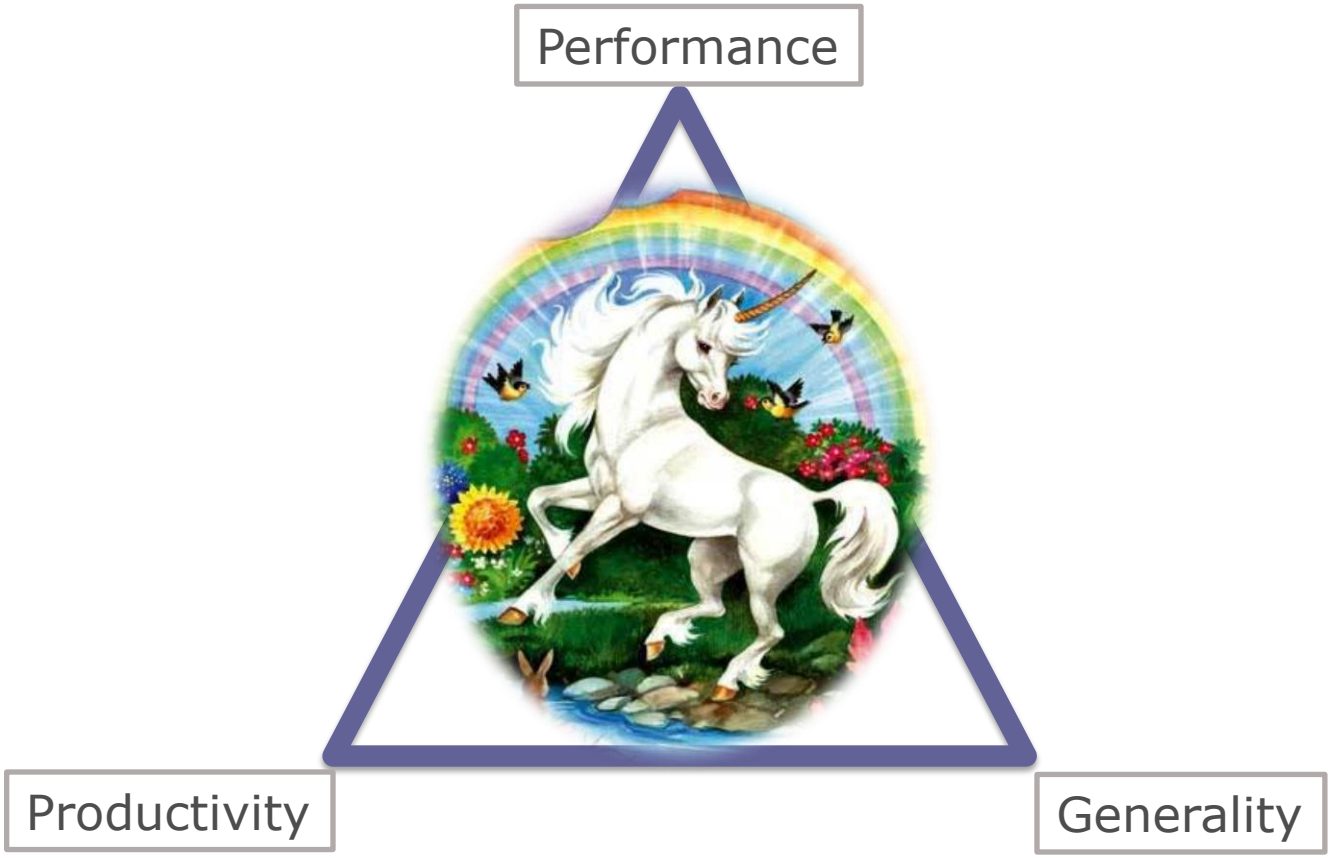
Altera FPGA

MPI
PGAS

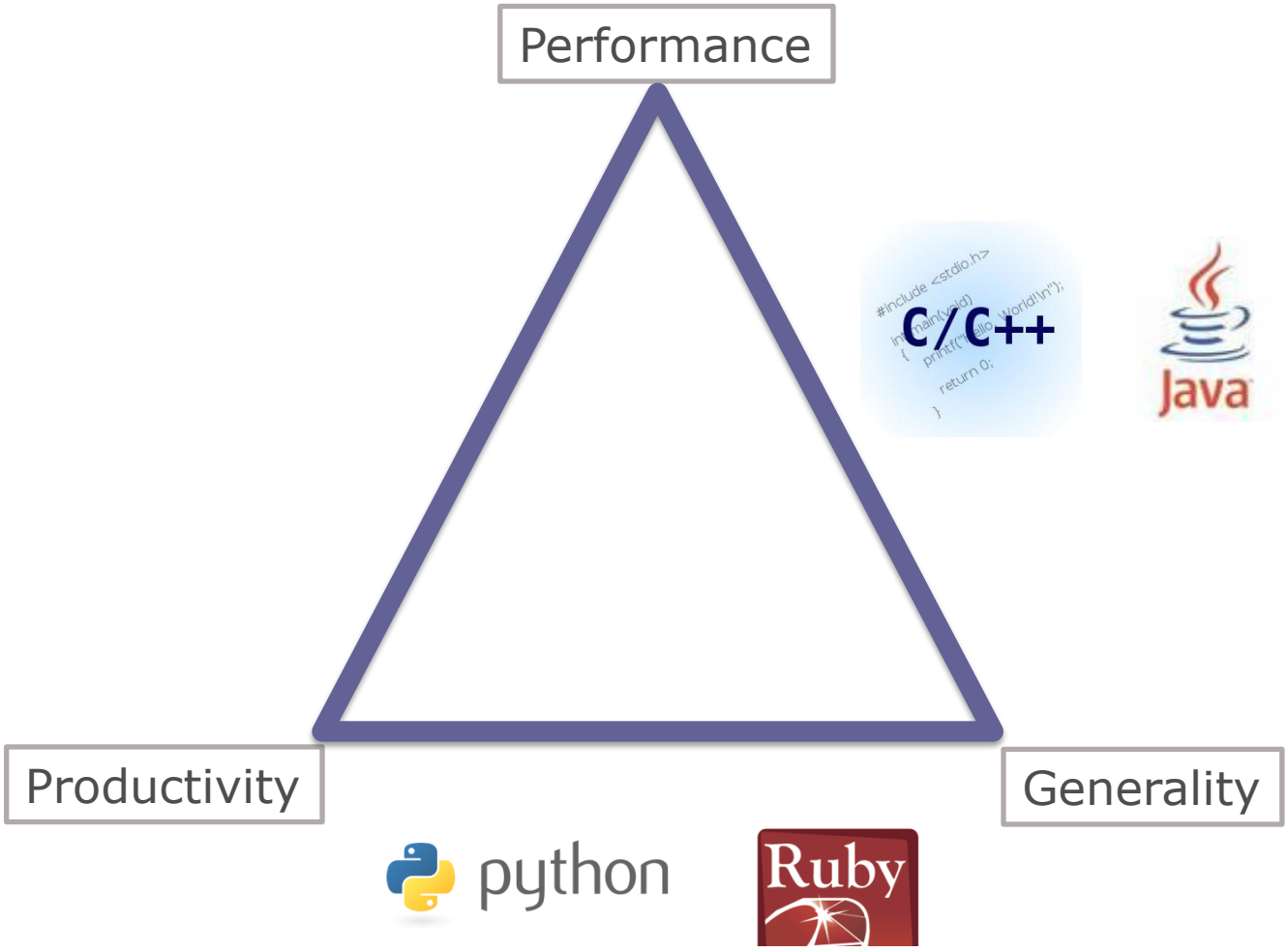


Cray Jaguar

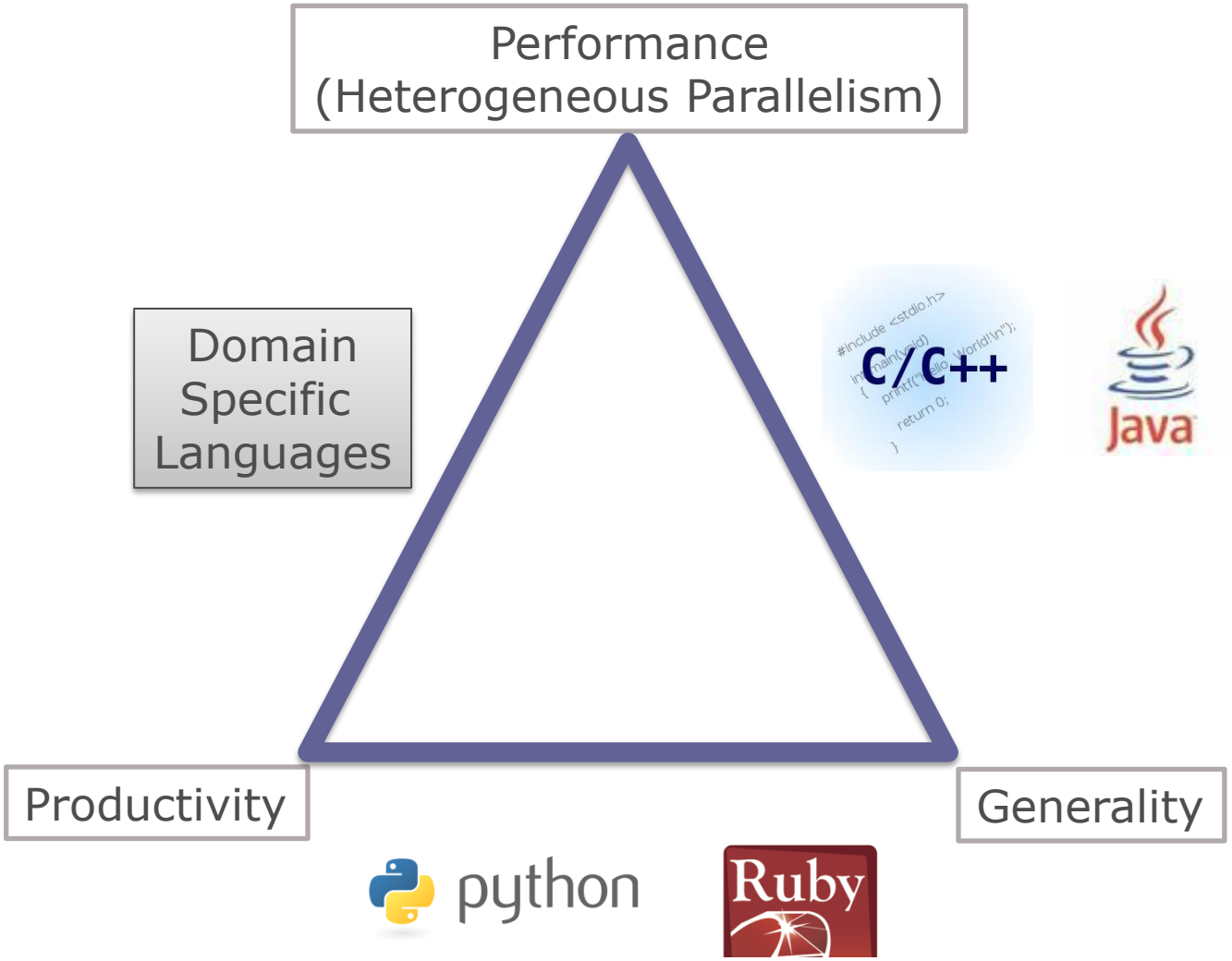
The Ideal Parallel Programming Language



Successful Languages



True Hypothesis ⇒ Domain Specific Languages



Domain Specific Languages

- Domain Specific Languages (DSLs)
  - Programming language with restricted expressiveness for a particular domain
  - High-level, usually declarative, and deterministic

DSL Benefits

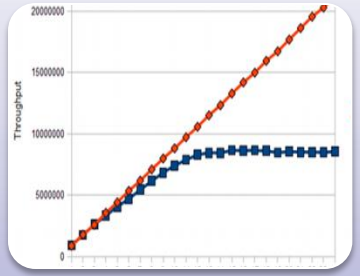
PERVASIVE PARALLELISM LABORATORY PPL

Productivity

- Shield average programmers from the difficulty of parallel programming
- Focus on developing algorithms and applications and not on low level implementation details

Performance

- Match high level domain abstraction to generic parallel execution patterns
- Restrict expressiveness to more easily and fully extract available parallelism
- Use domain knowledge for static/dynamic optimizations



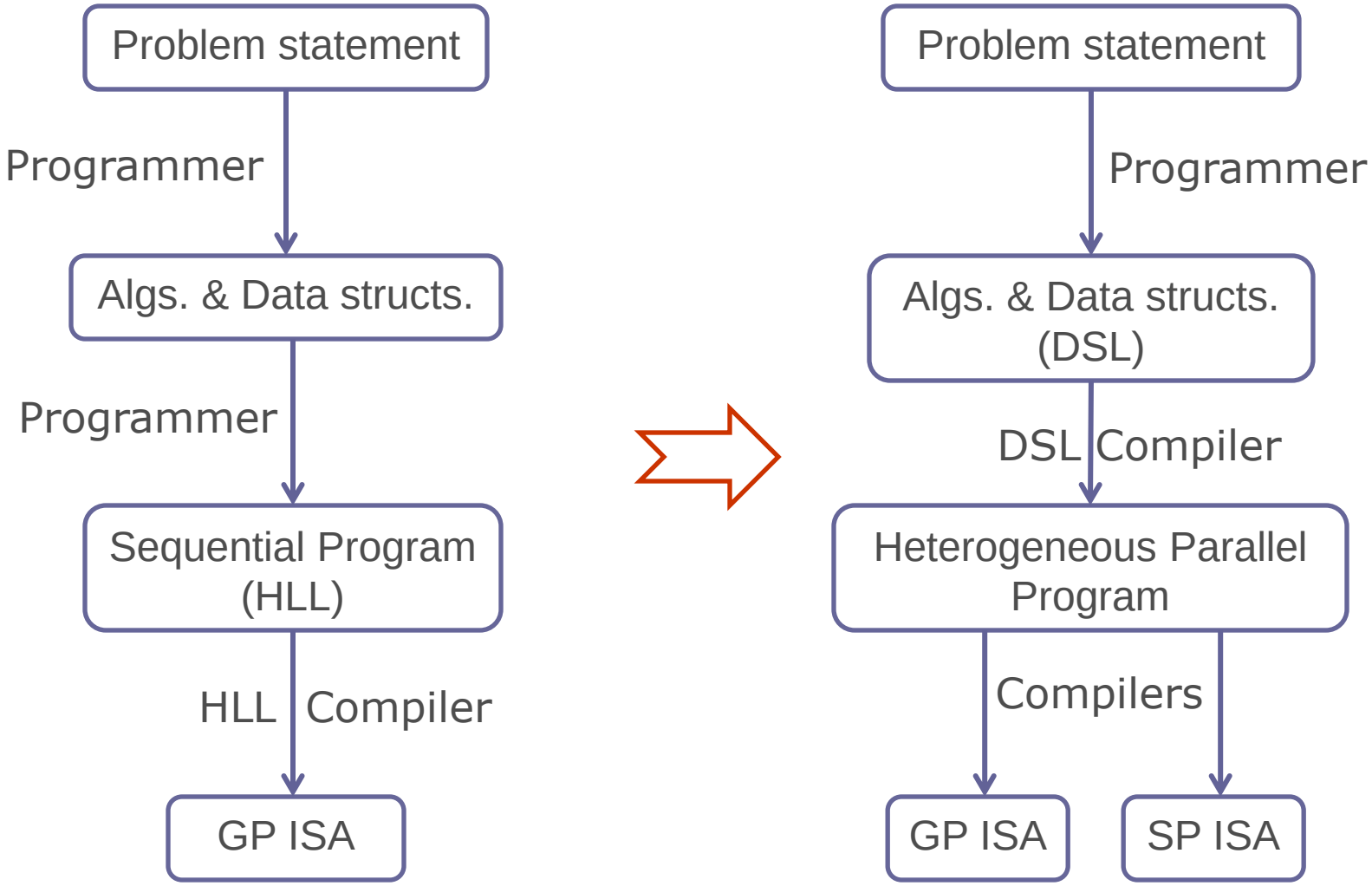
Portability and forward scalability

- DSL & Runtime can be evolved to take advantage of latest hardware features
- Applications remain unchanged
- Allows innovative HW without worrying about application portability

Our Approach: Three Views

- Little embedded languages
  - Domain abstractions improve productivity
  - Domains provide specific knowledge
- Smart libraries
  - Libraries that can compile/optimize themselves
  - Optimizations cross library call boundaries
  - Optimizations exploit domain specific knowledge
- Smart compilers
  - Raise abstraction-level of compiler optimization
  - Load and stores ⇒ Data structures
  - Language statements ⇒ Algorithms

Reinterpreting Levels of Abstraction



Bridging the Programmability Chasm

Applications

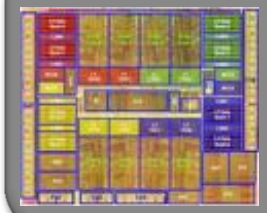
Scientific Engineering | Virtual Worlds | Personal Robotics | Data informatics

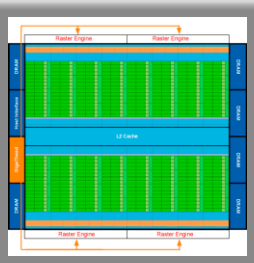
Domain Specific Languages

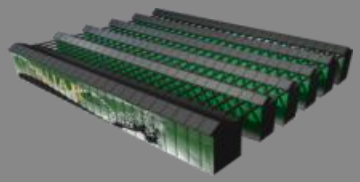
Statistics (R) | Physics (*Liszt*) | Data Analytics (OptiQL) | Graph Alg. (*Green Marl*) | Machine Learning (*OptiML*)

DSL Compiler | DSL Compiler | DSL Compiler | DSL Compiler | DSL Compiler

Heterogeneous Hardware

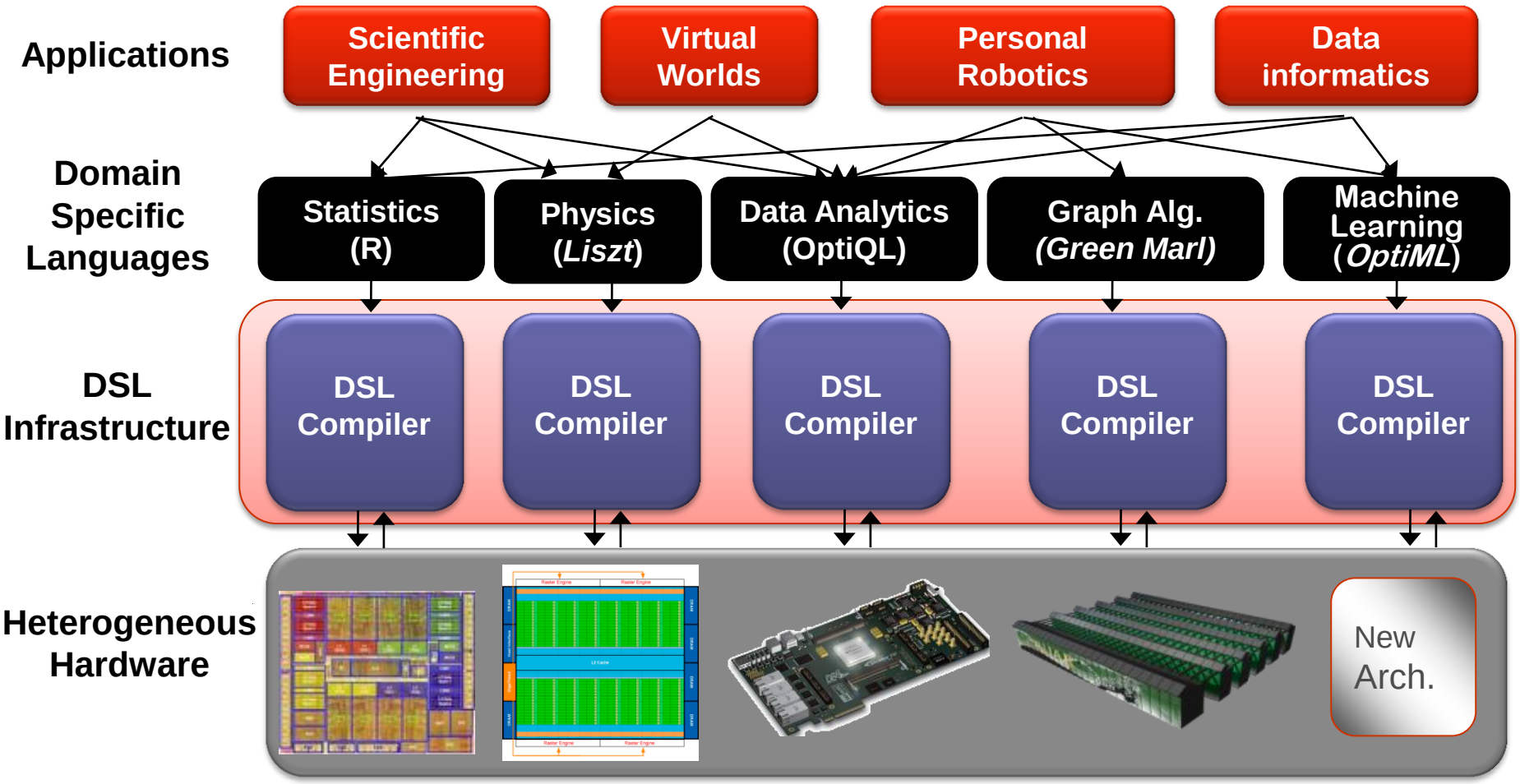




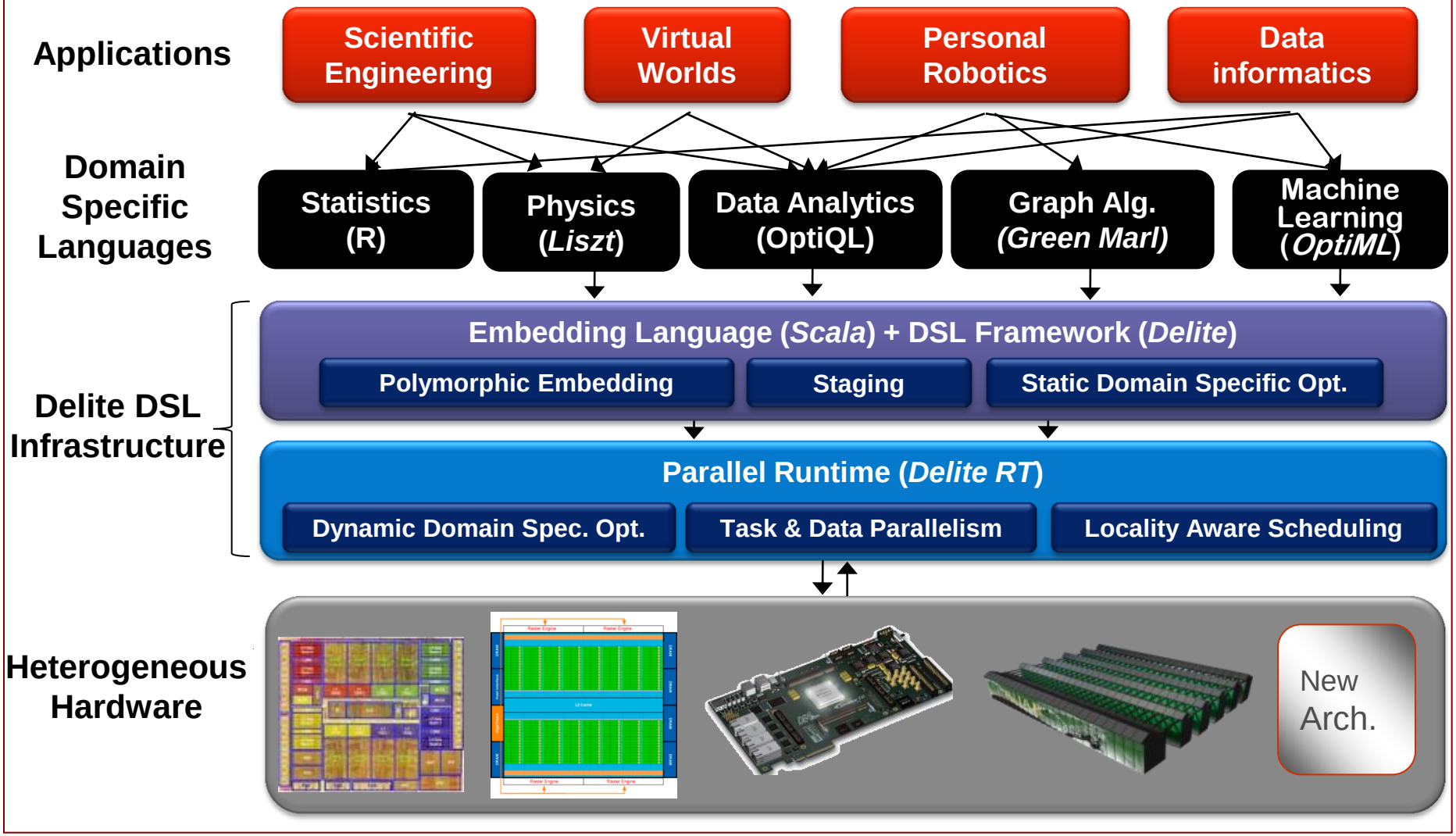


New Arch.

Common DSL Infrastructure



Delite DSL Framework



Agenda

- OptiML: A DSL for Machine Learning (Arvind Sujeeth)
- Liszt: A DSL for solving mesh-based PDEs (Zach DeVito)
- Green-Marl: A DSL for efficient Graph Analysis (Sungpack Hong)
- Scala Tutorial (Hassan Chafi)
- DSL Infrastructure Overview (Kevin Brown)
- High Performance DSL Implementation Using Delite (Arvind Sujeeth)
- Delite Status and Future Directions in DSL Research (Hassan Chafi)
- Wrap up (Kunle Olukotun)


PERVASIVE
PARALLELISM
LABORATORY
PPL

Tutorial Wrap Up

PERVASIVE PARALLELISM LABORATORY PPL

- Performance oriented DSLs
  - High productivity, performance and portability
  - Try out our DSLs (OptiML, Liszt, Green-Marl)
  - Develop your own DSLs: collaborate with domain experts
- Implementing DSLs with Delite
  - Embedded DSLs in Scala
  - Mapping to Delite IR
  - Domain specific optimizations
  - Optimizations for parallelism
  - Codegen for SMP and GPU, (Cluster)
  - Try out Delite, give us feedback
- Thanks for attending!