Deploying Accelerators At Datacenter Scale Using Spark

Di Wu and Muhuan Huang
University of California, Los Angeles,
Falcon Computing Solutions, Inc.
UCLA Collaborators: Cody Hao Yu, Zhenman Fang,
Tyson Condie and Jason Cong

SPARK SUMMIT 2016
DATA SCIENCE AND ENGINEERING AT SCALE
JUNE 6-8, 2016 SAN FRANCISCO

3x speedup in 3 hours

Accelerators in Datacenter

- CPU core scaling coming to an end
  - Datacenters demand new technology to sustain scaling
- GPU is popular, FPGA is gaining popularity
  - Intel prediction: 30% datacenter nodes with FPGA by 2020

Microsoft bing

IBM Cloudant

ALTERA now part of Intel

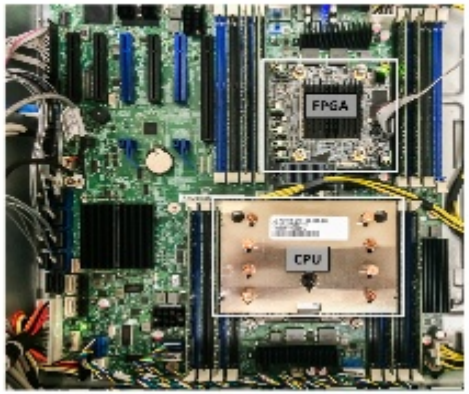
About us

- UCLA Center for Domain-Specific Computing
  - Expeditions in Computing program from NSF in 2009
  - Public-private partnership between NSF and Intel in 2014
  - http://cdsc.ucla.edu
- Falcon Computing Solutions, Inc.
  - Founded in 2014
  - Enable customized computing for big data applications
  - http://www.falcon-computing.com

What is FPGA?

- Field Programmable Gate Array (FPGA)
  - Reconfigurable hardware
  - Can be used to accelerate specific computations
- FPGA benefits
  - Low-power, energy efficient
  - Customized high performance

PCI-E FPGA
- IBM CAPI



FPGA in CPU socket
- Intel HARP

Problems of deploying accelerators in datacenters efficiently …

1. Complicated Programming Model

- Too much hardware specific knowledge
- Lack of platform-portability

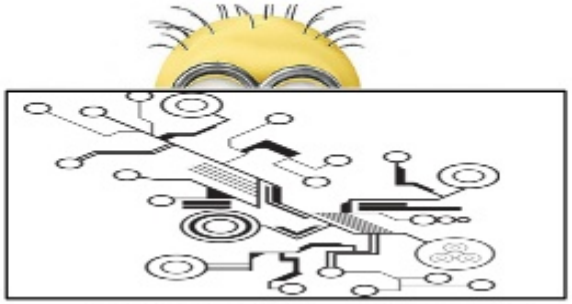
2. JVM-to-ACC data transfer overheads

- Data serialization/deserialization
- Additional data transfer

3. Accelerator management is non-trivial

Big-data application developer

How can I use your accelerator …?



Accelerator designer

Which cluster node has the accelerator …?

Does my accelerator work in your cluster …?

System administrator

More Challenges for FPGAs
4. Reconfiguration time is long

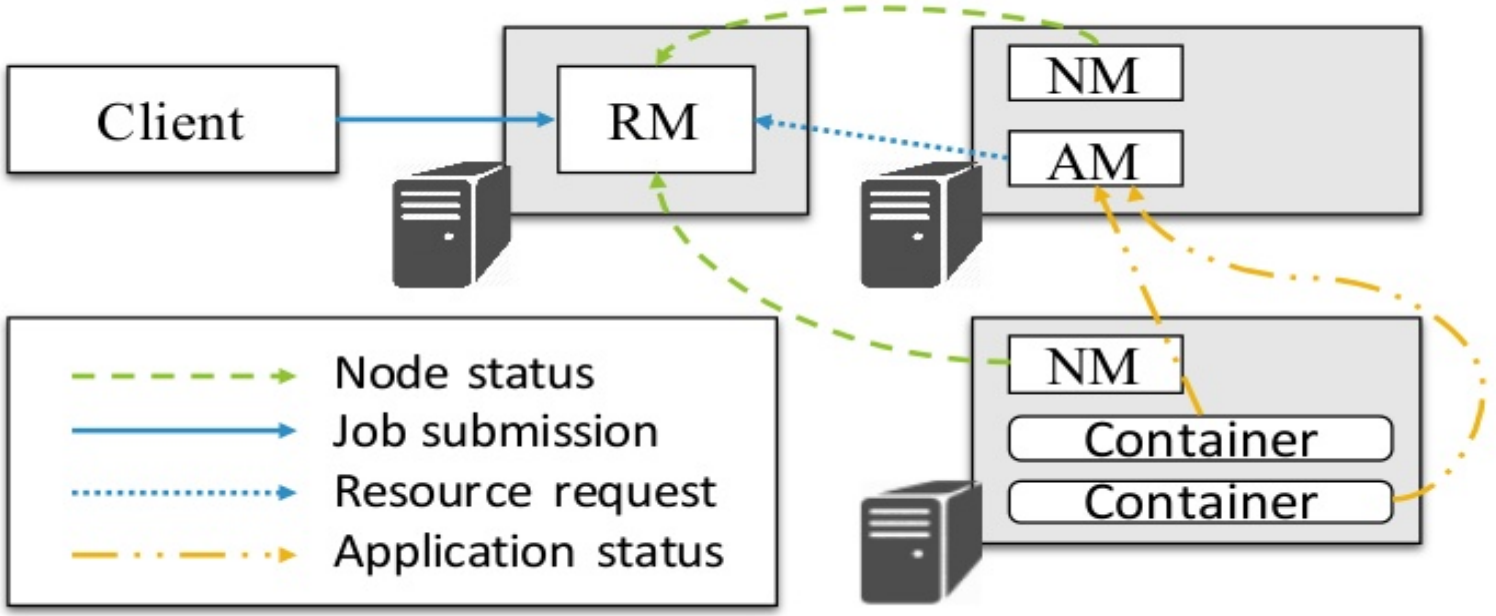
- Takes about 0.5 - 2 seconds
  - Transfer FPGA Binary
  - Reset the bits
  - …
- Naïve runtime FPGA sharing may slow down the performance by 4x

What we did

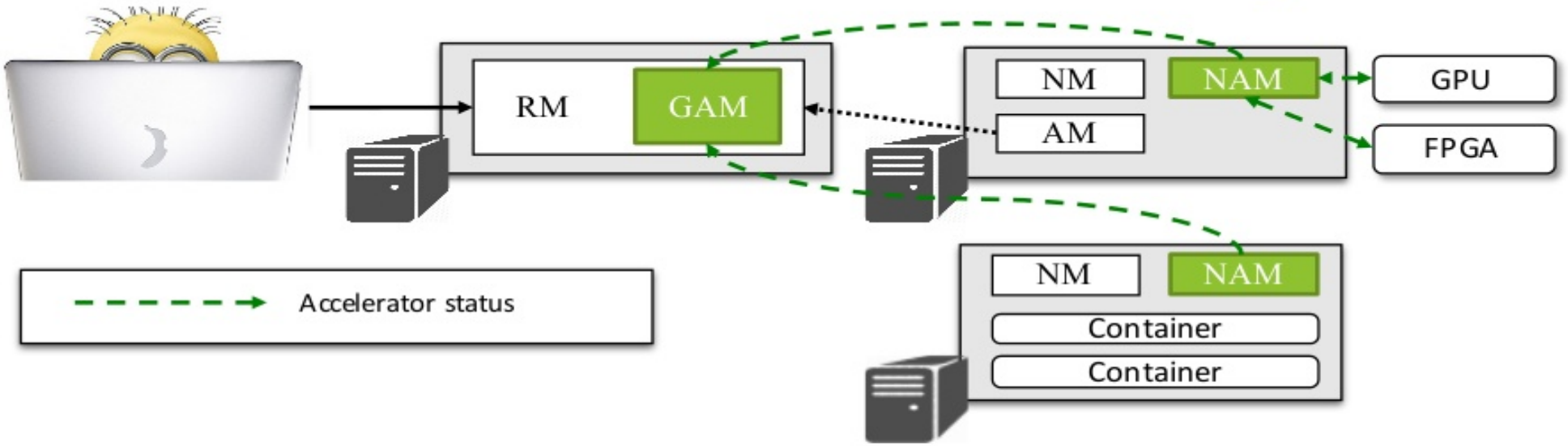
- Provide a better programming model:
  - APIs for accelerator developers
    - Easier to integrate into big-data workload, e.g. Spark and Hadoop
  - APIs for big-data application developers
    - Requires no knowledge about accelerators
- Provide an accelerator management runtime
  - Currently supports FPGAs and GPUs

➔ Blaze: a system providing Accelerator-as-a-Service

YARN Today


Client
RM
NM
AM
NM
Container
Container
Node status
Job submission
Resource request
Application status

Blaze: Accelerator Runtime System



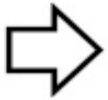
GAM — Global Accelerator Manager
accelerator-centric scheduling

NAM — Node Accelerator Manager
Local accelerator service management, JVM-to-ACC communication optimization

Details on Programming Interfaces and Runtime Implementation

Interface for Spark

```
val points = sc.textfile().cache
for (i <- 1 to ITERATIONS) {
  val gradient = points.map(p =>
    (1 / (1 + exp(-p.y*(w dot p.x)))
    - 1) * p.y * p.x
    ).reduce(_ + _)
  w -= gradient
}
```



```
val points = blaze.wrap(sc.textfile().cache)
for (i <- 1 to ITERATIONS) {
  val gradient = points.map(
     new LogisticGrad(w)
    ).reduce(_ + _)
  w -= gradient
}

class LogisticGrad(..)
     extends Accelerator[T, U] {
  val id: String = "Logistic"
}
```

blaze.wrap()


```
def compute():
  serialize data
  communicate with NAM
  deserialize results
```

Interface for Accelerators

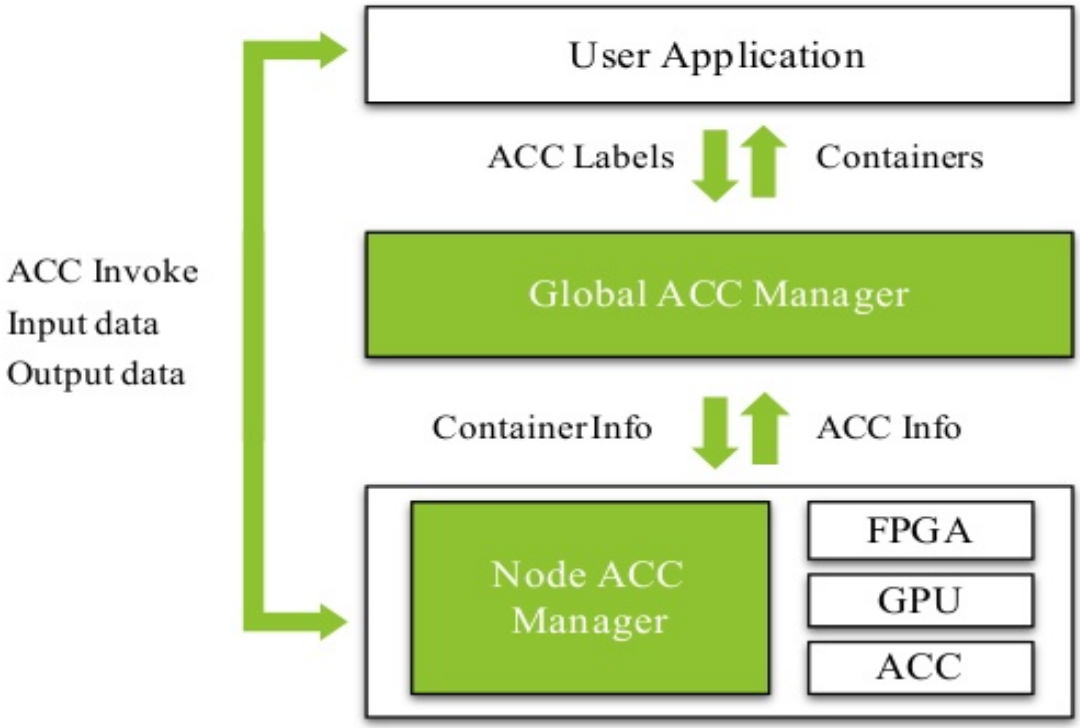
```
class LogisticACC : public Task {
// extend the basic Task interface
  LogisticACC(): Task() {;}
  // overwrite the compute function
  virtual void compute() {
    // get input/output using provided APIs
    // perform computation
  }
};
```

Interface for Deployment

- Managing accelerator services: through labels
- [YARN-796] allow for labels on nodes and resource-requests

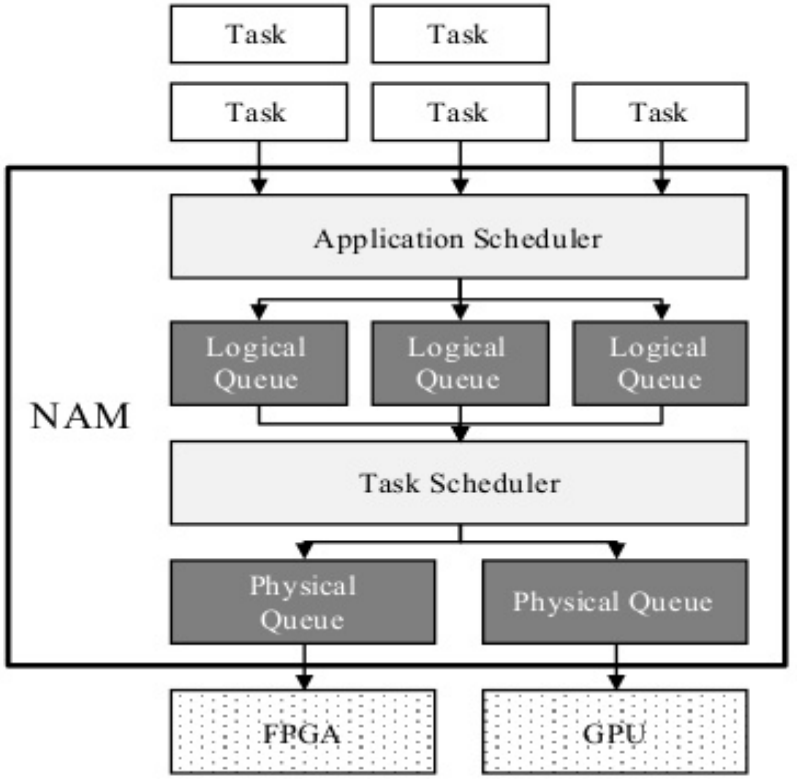
Putting it All Together

- Register
  - Interface to add accelerator service to corresponding nodes
- Request
  - Use acc_id as label
  - GAM allocates corresponding nodes



Accelerator-as-a-Service

- Logical Accelerators
  - Accelerator function
  - Services for applications
- Physical Accelerators
  - Implementation on a specific device (FPGA/GPU)

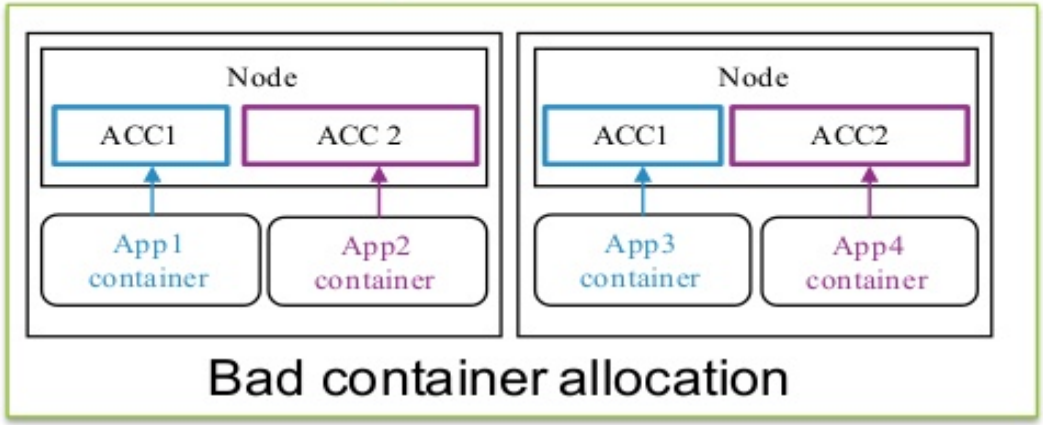


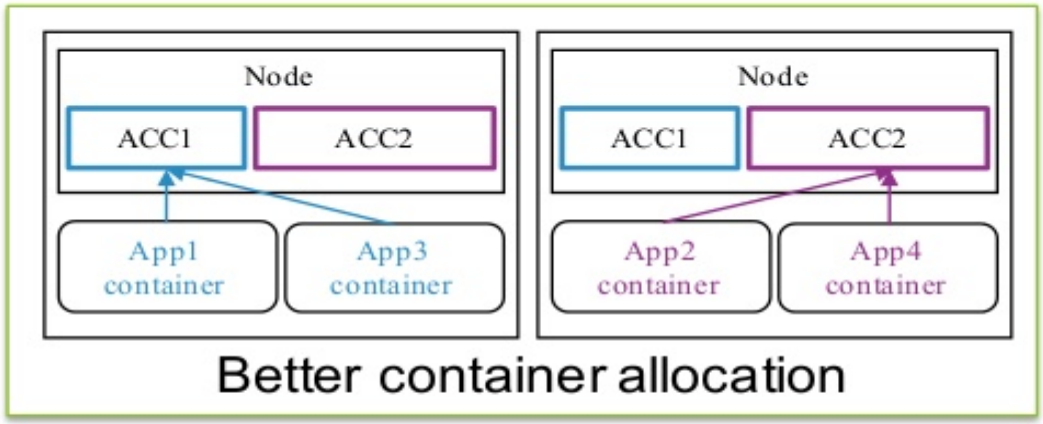
JVM-to-ACC Transfer Optimization

- Double-buffering / Accelerator Sharing
- Data caching
  - On GPU/FPGA device memory
- Broadcast

Global FPGA Allocation Optimization

- Avoid reprogramming
- GAM policy
  - Group the containers that need the same accelerator to the same set of nodes





Programming Efforts Reduction

Lines of Code	Accelerator Management
Logistic Regression (LR)	325 → 0
Kmeans (KM)	364 → 0
Compression (ZIP)	360 → 0
Genome Sequency Alignment (GSA)	896 → 0

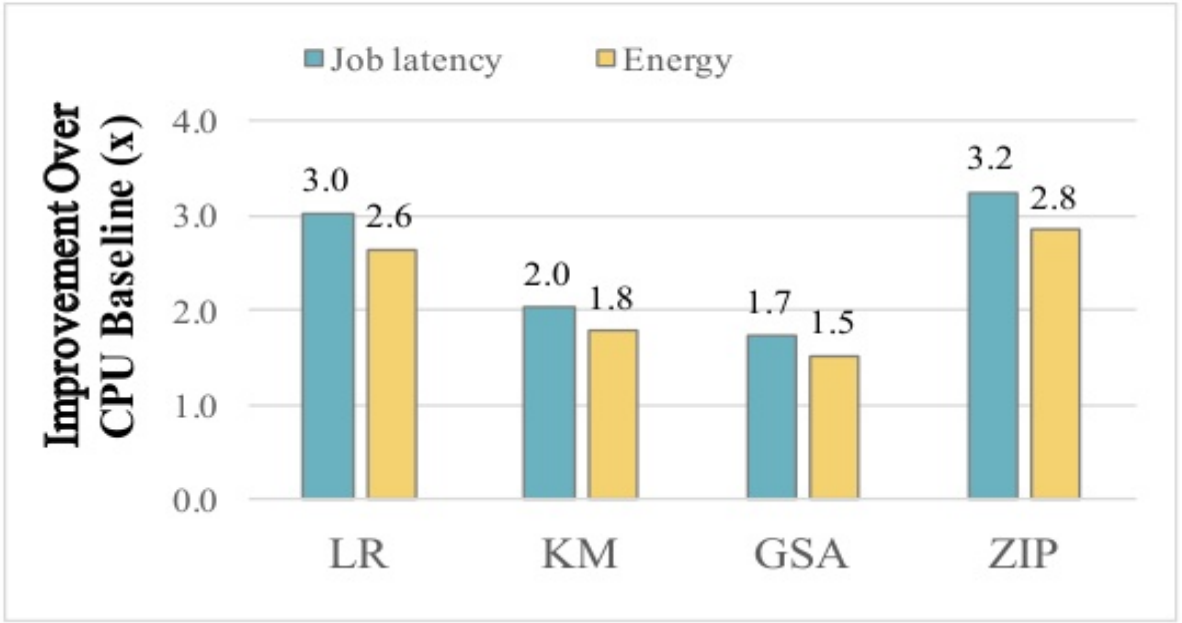
Heterogeneous Clusters

- CDSC and Falcon Clusters
  - Low-power GPUs
  - PCI-E FPGAs
- Workloads
  - Genome sequencing
  - Machine learning

System Performance and Energy Efficiency

1.7x ~ 3.2x Speedup

1.5x ~ 2.8x Energy reduction

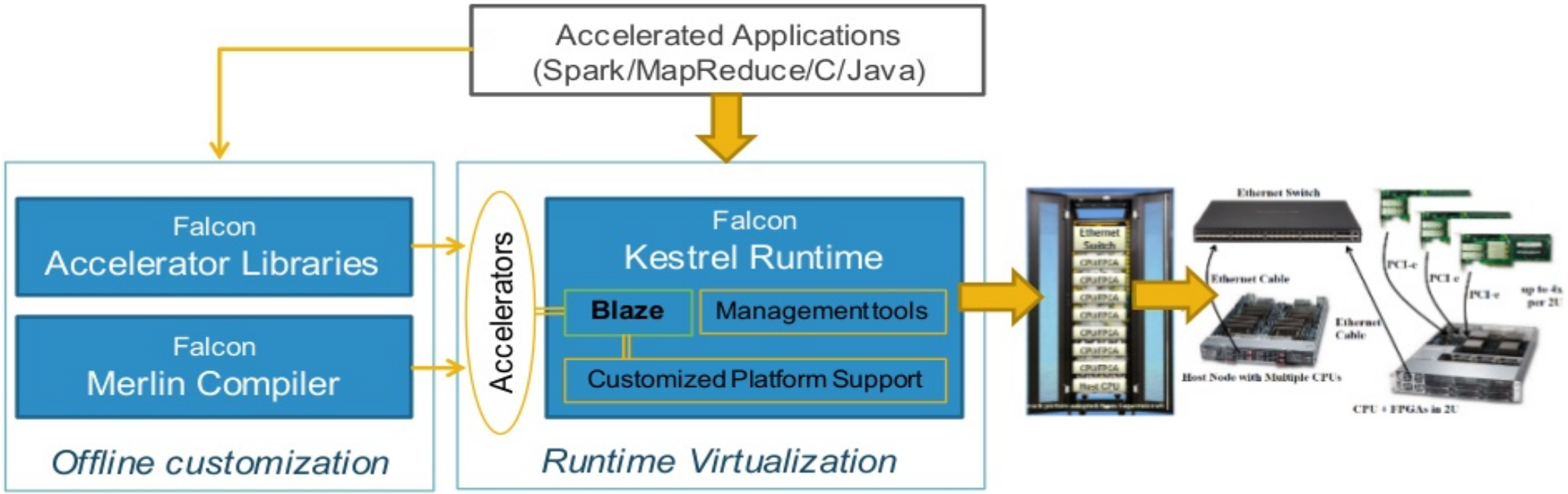


DEMO

Take Away

- Accelerator deployment can be made easy
- FPGA requires special considerations
- Key to efficiency is JVM-to-ACC overheads
  - Looking for new ideas
- Blaze is an open-source project
  - Looking for collaboration

About Falcon Computing



We thank our sponsors:

- NSF/Intel Innovation Transition Grant awarded to the Center for Domain-Specific Computing
- Intel for funding and machine donations
- Xilinx for FPGA board donations

THANK YOU.

Di Wu, allwu@cs.ucla.edu
Muhuan Huang, mhhuang@cs.ucla.edu

Blaze: http://www.github.com/UCLA-VAST/blaze

Center for Domain-Specific Computing: http://cdsc.ucla.edu
Falcon Computing Solutions, Inc.: http://www.falcon-computing.com

SPARK SUMMIT 2016
DATA SCIENCE AND ENGINEERING AT SCALE
JUNE 6-8, 2016 SAN FRANCISCO