Oak Ridge National Laboratory

OLCF Training

# AI on Frontier

**Junqi Yin**

Analytics and AI Methods at Scale

Feb 16, 2023

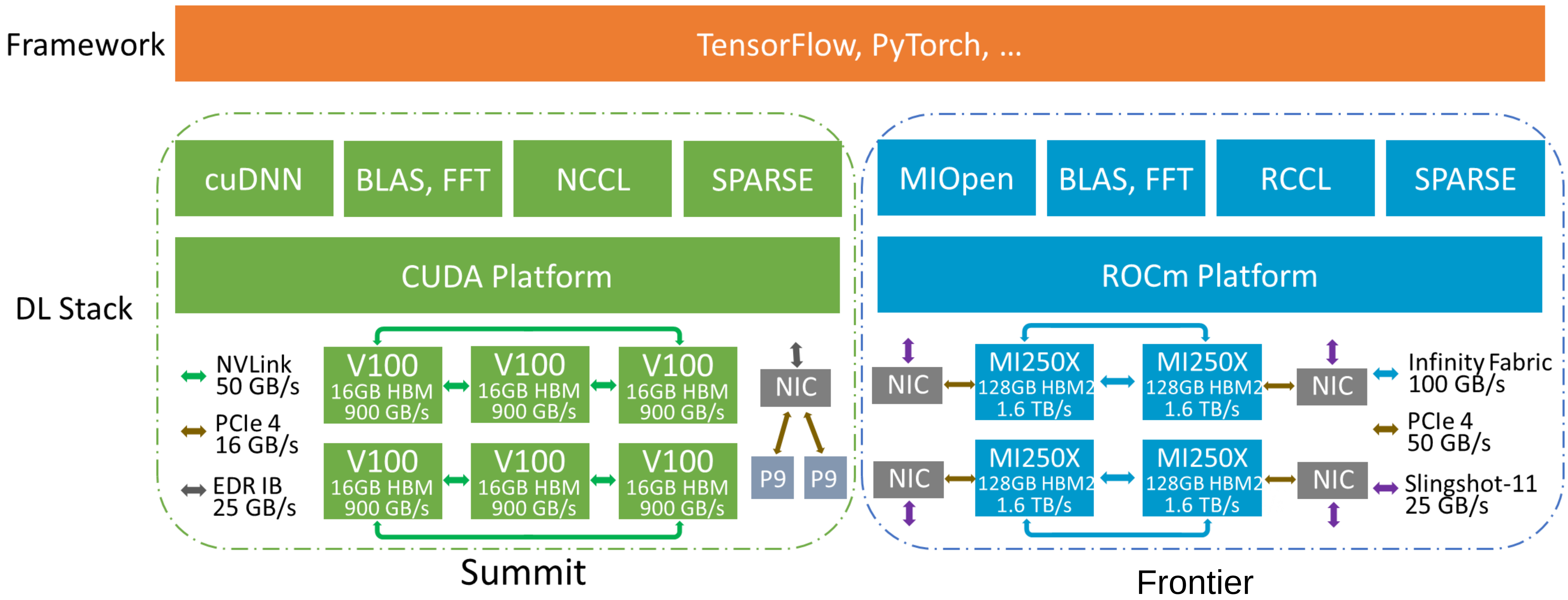


U.S. DEPARTMENT OF ENERGY

# Outline

- Frontier DL Environment
- Prelimenary performance numbers
    - Kernels: GEMM/CONV/LSTM
    - Models: CNN/RNN
    - Applications: ResNet50, STEMDL
- Simulation-ML integration

# Deep Learning Stacks




- Most DL codes work on Frontier without changes

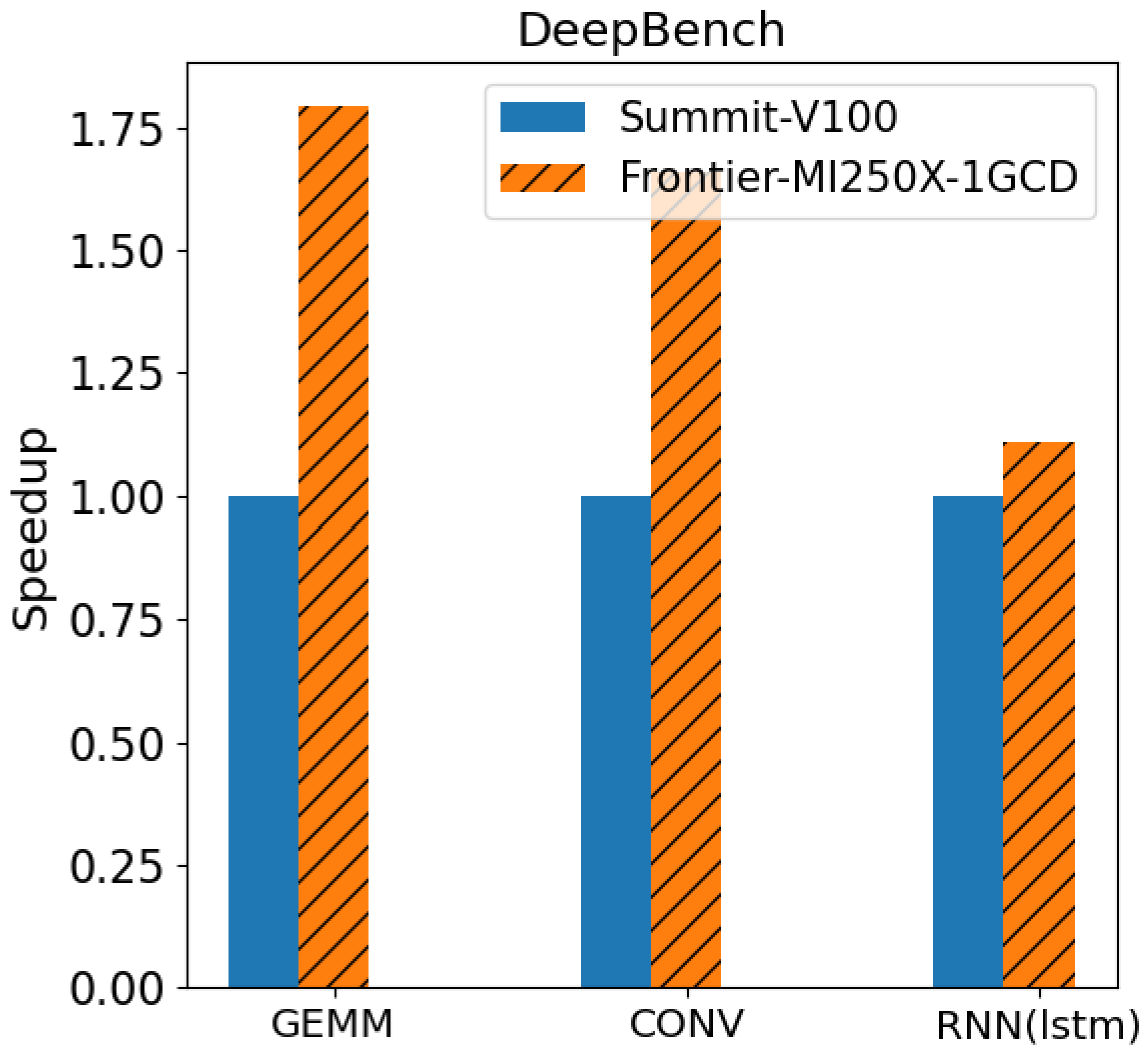
# Frontier DL Environment

- What's working?
  - TensorFlow and PyTorch
  - Third-party libraries
    - Cray DL-plugin
    - Horovod
    - DeepSpeed
    - PyG ...
- Peculiarities
  - rocm-smi
  - MIOpen cache
  - RCCL + libfabric

- Resources: https://github.com/ROCmSoftwarePlatform

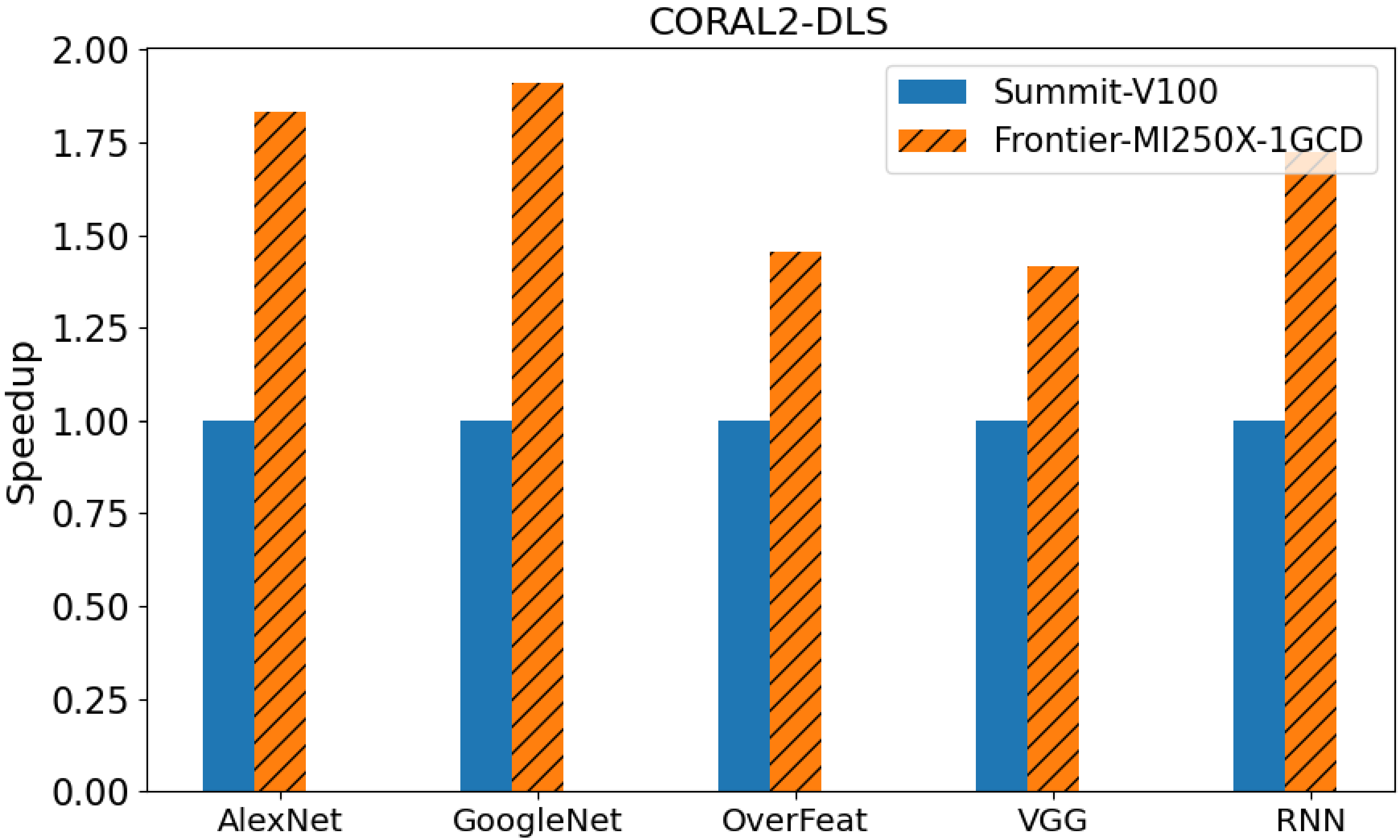
# Performance baselines: Kernels

- Kernel Ops (fp32)
  - GEMM ~ 1.7x
  - CONV ~1.6x
  - LSTM ~1.1x



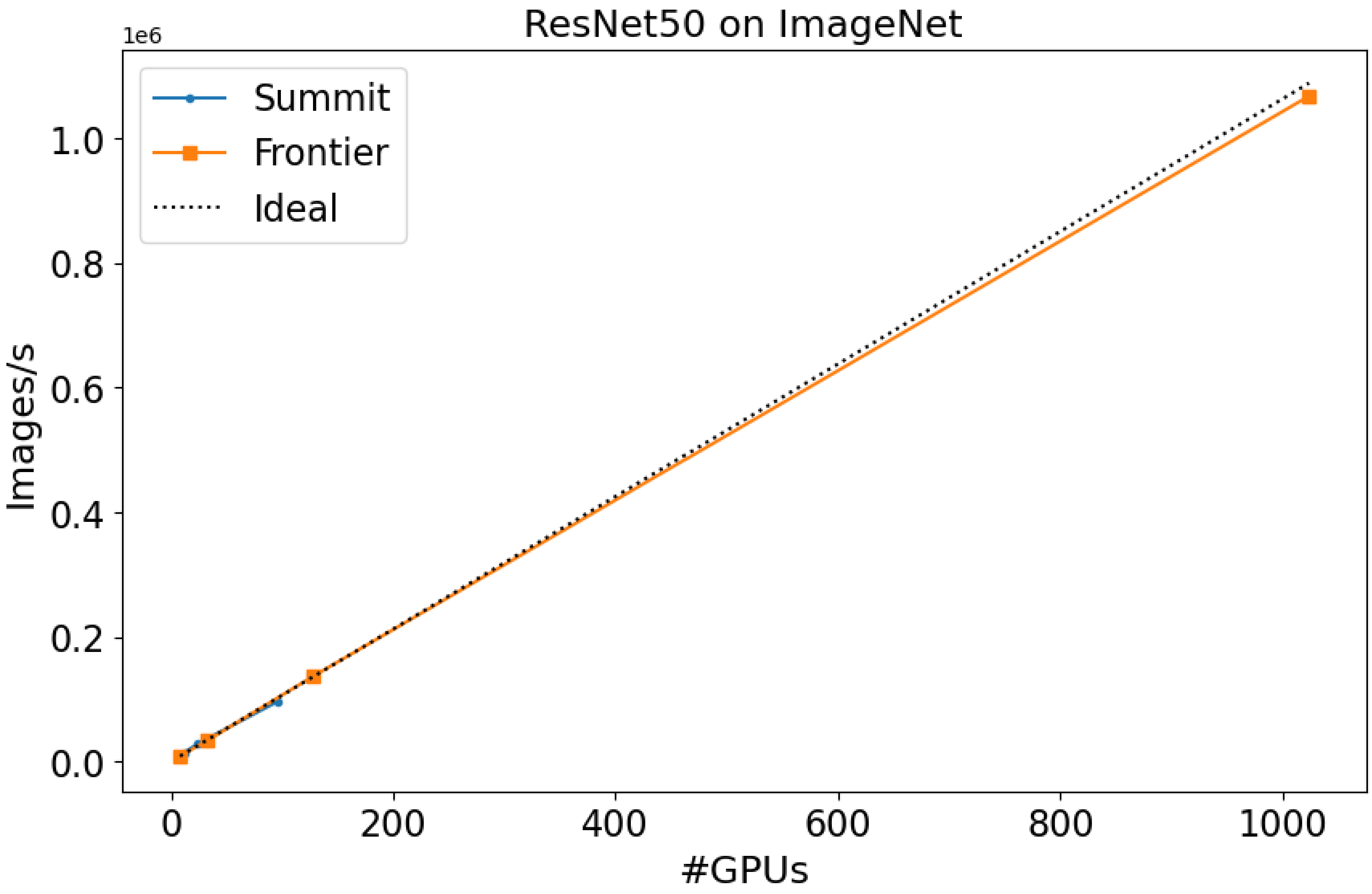
# Performance baselines: Models

- CNN (fp32)
  - AlexNet
  - GoogleNet
  - OverFeat
  - VGG
- RNN



# Performance baselines: Apps

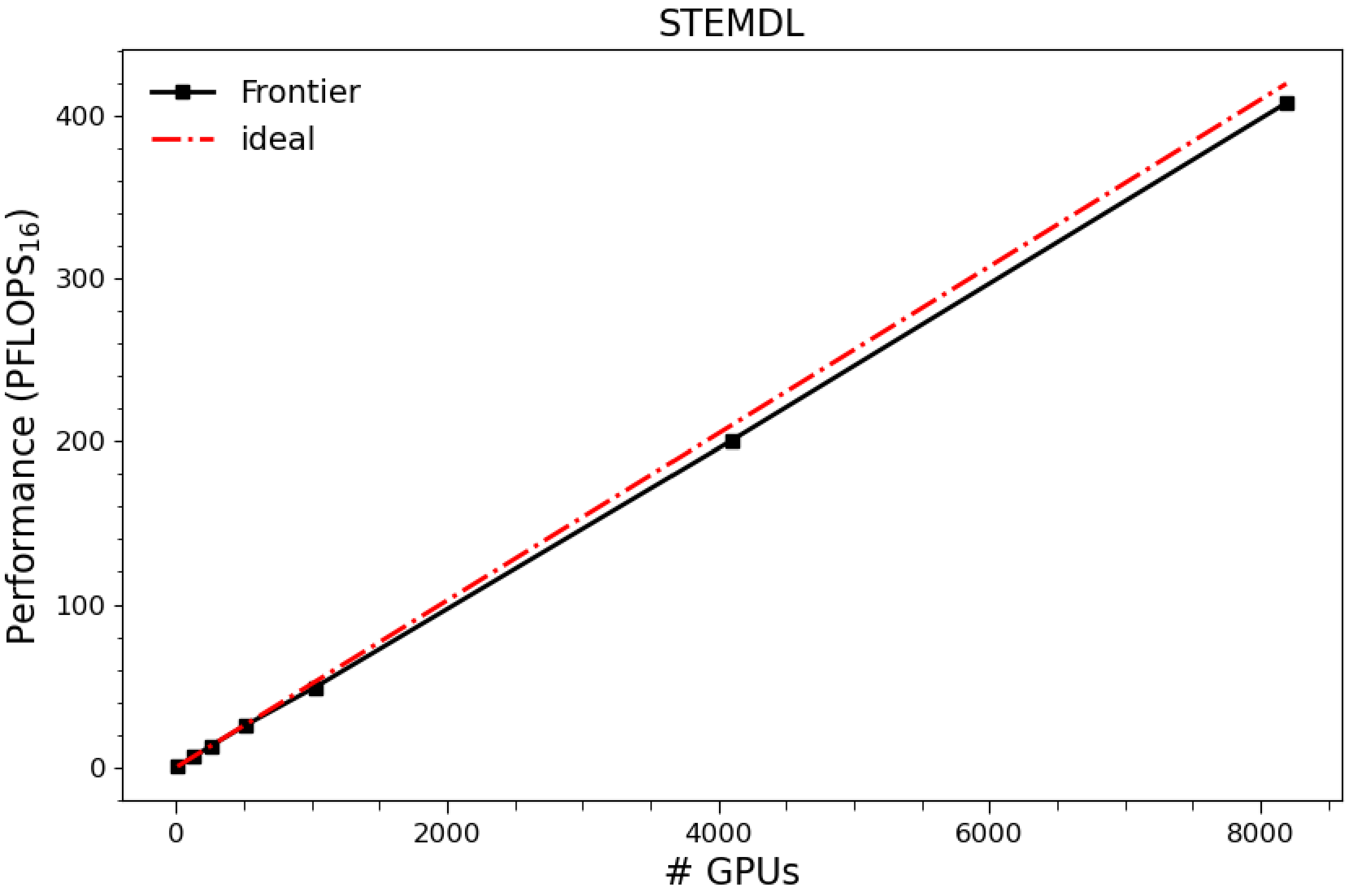
- ResNet50
  - Mixed
  - ~1.0x per GCD
  - 98% at 1024



# Performance baselines: Apps

- STEMDL
  - Tiramisu network
  - 220M parameters
  - 97% at 8192 GPUs

Accelerating Collective Communication in Data Parallel Training across Deep Learning Frameworks. USENIX NSDI’22,  2022

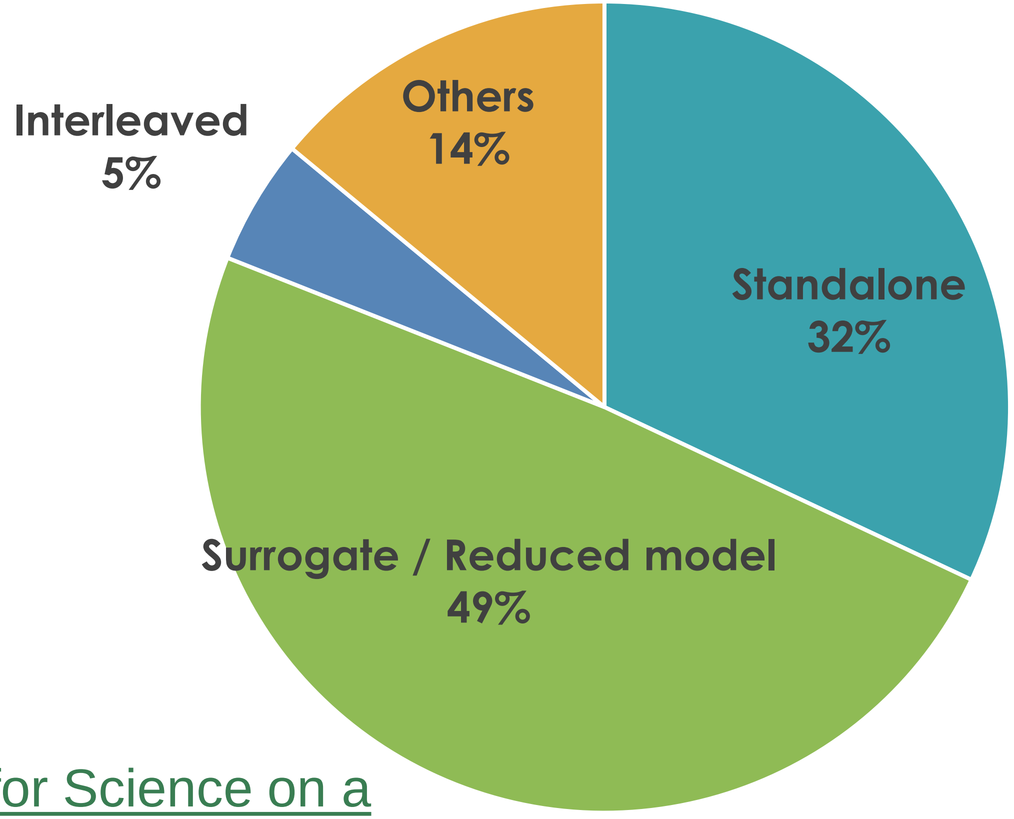


# Simulation-ML Integration

- Best of both worlds: FP64 simulation + FP16 modeling

- Common use cases
  - Surrogate modeling
  - Reduced model
  - Interleaved

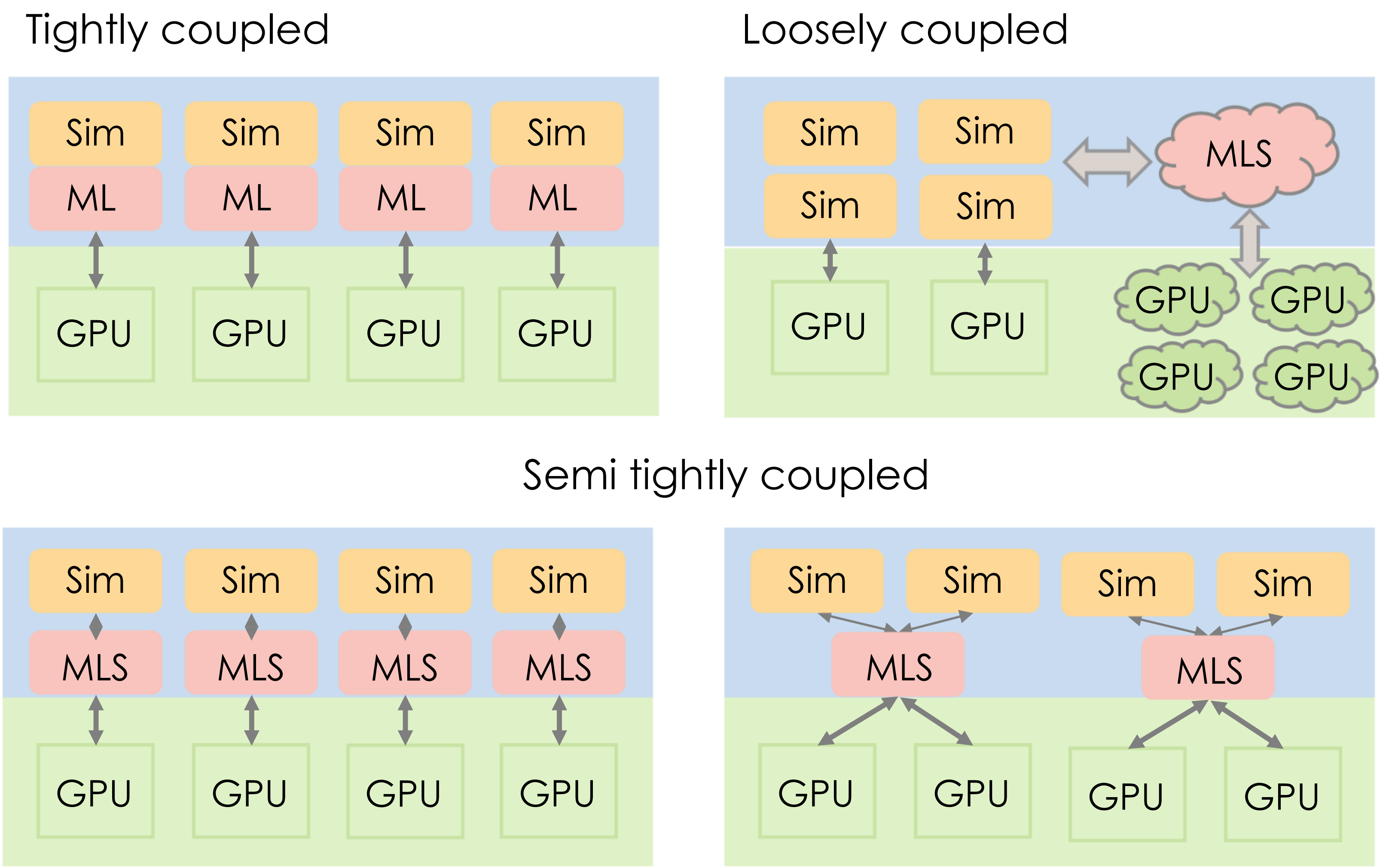
OLCF (2019 - 2021)
662 projects (147 INCITE, 72 ALCC, 62 ECP, DD)




[Learning to Scale the Summit: AI for Science on a Leadership Supercomputer, IPDPSW 2022]

# Simulation-ML Integration

- Tightly coupled
  - Single executable
  - one-to-one
- Loosely coupled
  - Different machines
  - many-to-many
- Semi-tightly
  - Separate executables
  - 1-to-1, many-to-many



# Simulation-ML Integration

| Method | Pro | Con |
|---|---|---|
| Framework C++ API (TensorFlow/PyTorch C++) | • Portable<br>• Better latency<br>• Easy to deploy | • Not flexible |
| Framework Server (TensorFlow Serving/TorchServe) | • Flexible<br>• Better throughput<br>• Options to deploy | • High maintenance |
| Third-party API (SmartRedis/RedisAI) | • Easy integration<br>• More functionality<br>• Model support | • Portability |

# TensorFlow C++

- Assume model in TF *SavedModel* format
- Link with *libtensorflow_cc.so*
- Support *half*, *uint8*, …

**Implementation 1** Use TensorFlow C++ API

```
function LOADMODEL(string ModelPath)
    tensorflow::SessionOptions SessOpt
    tensorflow::RunOptions RunOpt
    tensorflow::SavedModelBundle Model          ▷ Get model path, and setup session and run options
    tensorflow::LoadSavedModel(SessOpt, RunOpt,
        ModelPath, {"serve"}, &Model)           ▷ load pre-trained model
    return Model
end function

function PREDICT(tensorflow::SavedModelBundle Model, tensorflow::Tensor Inputs)
    tensorflow::Tensor Inputs                   ▷ input tensor
    vector<tensorflow::Tensor> Outputs
    string InputNode
    string OutputNode                           ▷ input and output nodes

    Model.GetSession().Run(data, {OutputNode}, {}, &Outputs)
    return Outputs
end function

function ENERGY
    for i ← 0, Nelements do
        Model_i ← LOADMODEL(Path_i)
    end for

    for i ← 0, Nelements do                     ▷ Add energy by element species
        tensorflow::TensorShape DataShape
        tensorflow::Tensor Inputs(tensorflow::DT_UINT8, DataShape)
        for j ← 0, Nneighbors do
            Inputs ← Atom[i][j]
        end for
        Outputs = PREDICT(Model_i,Inputs)
        Energy ← Outputs
    end for
    return Energy
end function
```

# TensorFlow Serving

- Launch server: *tensorflow_model_server --port --model_config_file*
- Support grpc & http

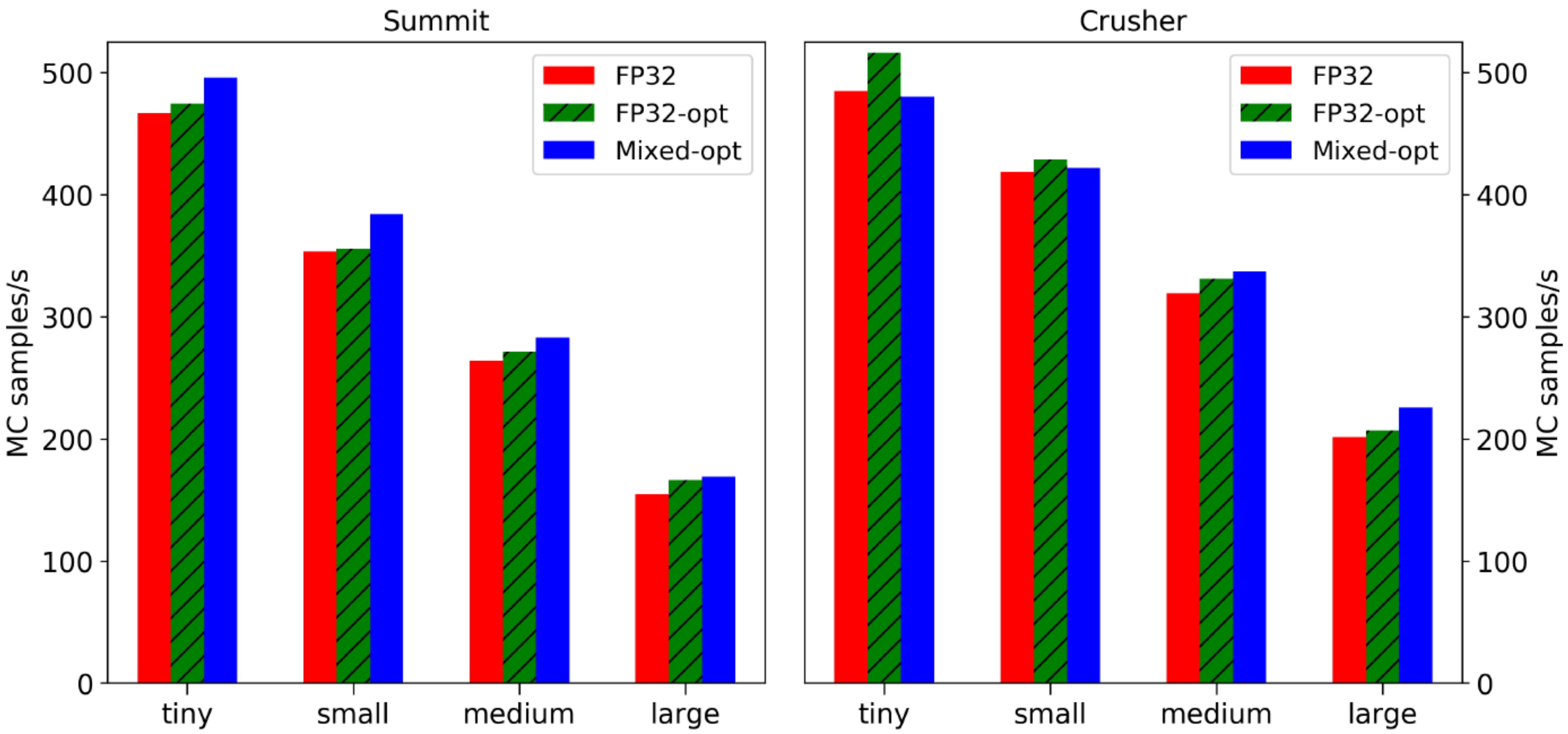
**Implementation 2** Use TensorFlow Serving

```
function ENERGY
    Host ← server_ip : port
    for i ← 0, Nelements do
        Stub_i ← tensorflow::serving::PredictionService::NewStub(
grpc::CreateChannel(Host, grpc::InsecureChannelCredentials()))
    end for

    for i ← 0, Nelements do                          ▷ Add energy by element species
        tensorflow::serving::PredictRequest predictRequest
        tensorflow::serving::PredictResponse Outputs
        grpc::ClientContext context
        predictRequest.mutable_model_spec().set_name(Model_i)
        predictRequest.mutable_model_spec().set_signature_name("serving_default")
        for j ← 0, Nneighbors do
            predictRequest.mutable_inputs() ← Atom[i][j]
        end for
        Stub_i.Predict(&context, predictRequest, &Outputs)
        Energy ← Outputs
    end for
    return Energy
end function
```

# Simulation-ML Integration

- 1.2x per GCD over V100




[Strategies for Integrating Deep Learning Surrogate Models with HPC Simulation Applications, IPDPSW 2022]

Questions ?

yinj@ornl.gov