

Integrating AI and Simulations Part 1

Bethany Lusch & Saumil Patel
Argonne Leadership Computing Facility

SDL Workshop
October 6, 2022

Pro-tip: grab an interactive node ahead of the demo

From: [sdl_workshop/couplingSimulationML/ML_PythonC++_Embedding/ThetaGPU_OCCA/README.md](#)

1. From the theta login node, please Login to a ThetaGPU service node

```
ssh thetagpusn1
```

2. Request an interactive session on an A100 GPU

```
qsub -A SDL_Workshop \  
      -q training-gpu \  
      -I \  
      -n 1 \  
      --attrs filesystems=home,grand,eagle \  
      -t 60
```

Why Couple AI and Simulation?

Example Use Cases

- Use machine learning to decide which simulations to launch
- Replace part of a simulation with a machine learning surrogate
- Reduce I/O, e.g. can't save simulations, so train while they are running
- Use ML to control the simulations, select simulation parameter

Ways to Couple AI & Simulation

Example Modes

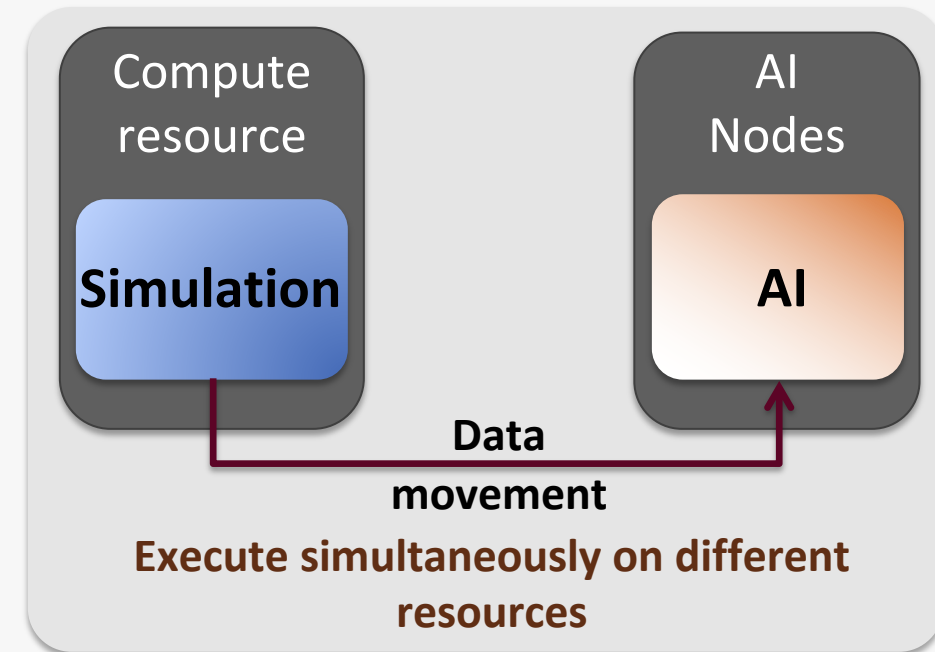
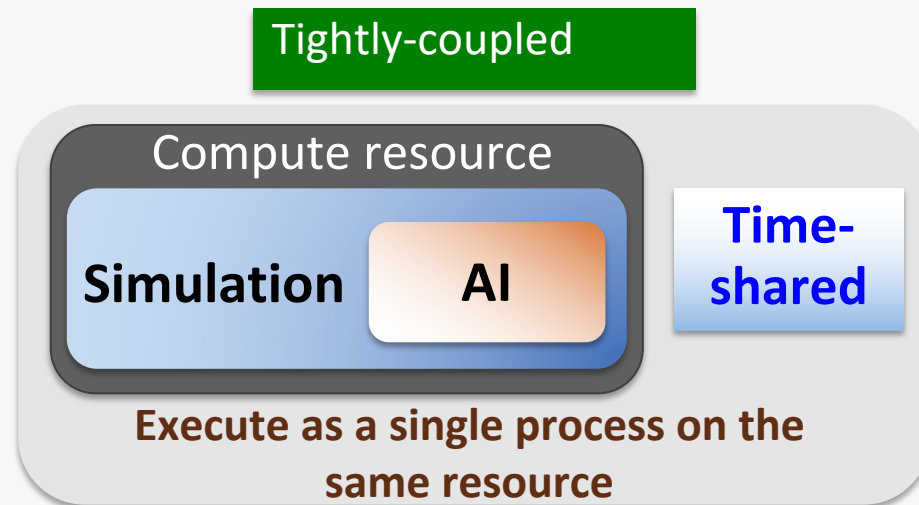
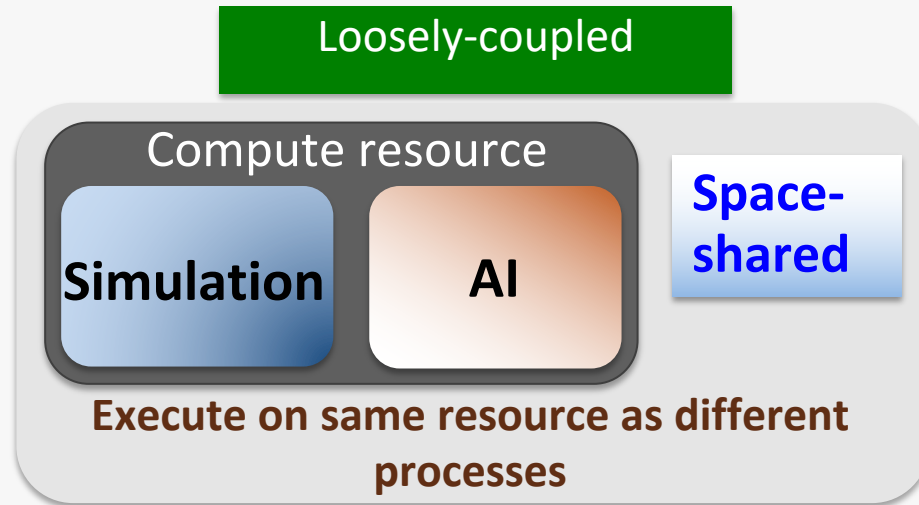


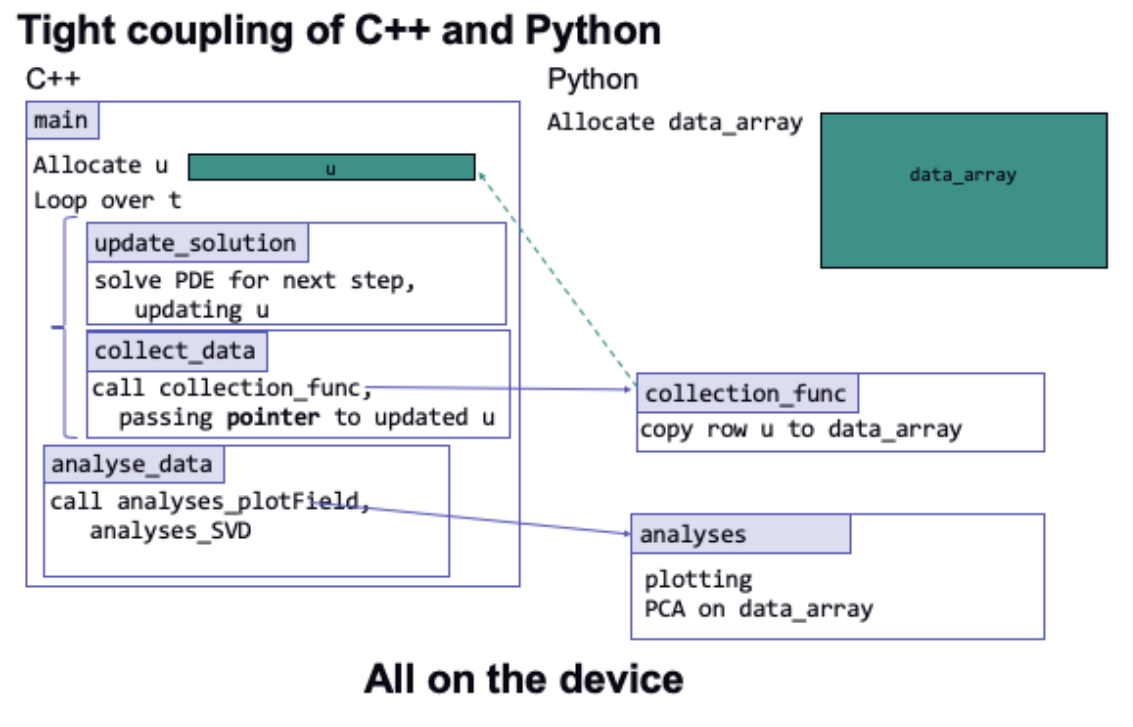
Figure adapted from Venkat Vishwanath

“A terminology for in situ visualization and analysis systems” by Childs et al. 2020

Background for Demo 1

Demo 1: tight coupling between Python and C++

- For more details, see the README
- We are able to pass data back and forth between Python & C++ without copying, and remaining on the device
 - Using Python and Numpy C APIs, plus
 - Using OCCA (a performance-portability abstraction layer)
- By integrating Python, we are able to use convenient Python libraries not available in C++



Bonus Slides: extra detail

Choose best simulations to launch

- Active learning: ones that improve ML model
- Or “experiment design”
- Diverse set?
- Ones likely to improve quantity of interest?
- If ML model degrading, collect more data?
- Sample space more effectively

Example:

“Machine Learning Inter-Atomic Potentials Generation Driven by Active Learning” Sivaraman, et al.

<https://arxiv.org/abs/1910.10254>

<https://github.com/argonne-lcf/active-learning-md>

Surrogate modeling

Replace *part* of simulation with ML surrogate model

- Expensive part?
- Inaccurate part?

ML model output fed back into rest of simulation

Example:

“A turbulent eddy-viscosity surrogate modeling framework for RANS simulations”

By Maulik, et al.

<https://doi.org/10.1016/j.compfluid.2020.104777>

<https://github.com/argonne-lcf/TensorFlowFoam>

Reduce I/O

- Apply ML model to save compressed simulation results
- Train online during simulation (skip I/O bottleneck)
- *In situ* analysis giving feedback on simulations before completed
 - Need to adjust something?

Example:

“In Situ Compression Artifact Removal in Scientific Data Using DeepTransfer Learning and Experience Replay” by Madireddy, et al.

<https://doi.org/10.1088/2632-2153/abc326>

Control Simulation with ML

- Select simulation parameters
- Select numerical scheme

Example:

“Distributed Deep Reinforcement Learning for Simulation Control”

By Pawar & Maulik

<https://arxiv.org/pdf/2009.10306.pdf>

https://github.com/Romit-Maulik/RLLib_Theta/

Other Use Cases

- Data assimilation
- Augmenting simulation with ML closure/discrepancy model
- Solver as part of ML loss function

How Close is the Coupling?


Proximity

- On node
- Off node, same computing resource
- Distinct computing resource

Data Access

- Direct: share same logical memory space
- Indirect: distinct logical memory
- Either way: need to synchronize data

May or may not require copy



“A terminology for in situ visualization and analysis systems” by Childs et al. 2020

Division of Execution

- Space Division (Different physical compute resources)
 - Can allocate appropriate resource to each
 - But need to keep both utilized & transfer data
- Time Division (Some compute resources alternate simulation vs. AI)
 - Less or no synchronization & data transfer
 - But always blocking one or the other

“A terminology for in situ visualization and analysis systems” by Childs et al. 2020