Novel Architecture Infrastructure for Addressing Future HPC Data Challenges

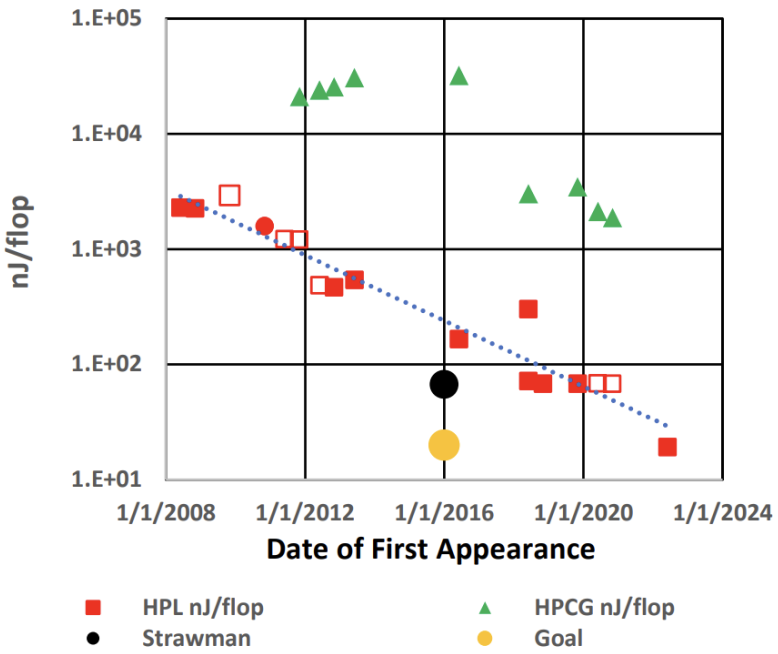
Jeffrey Young, PhD · Rogues Gallery Director, Principal Research Scientist · Partnership for Advanced Computing Environments

Durham University SciComp 2025 Seminar
February 20th, 2025

* Many thanks to Aaron Jezghani, SRS, PACE and Sara Karamati and Rich Vuduc for slide content

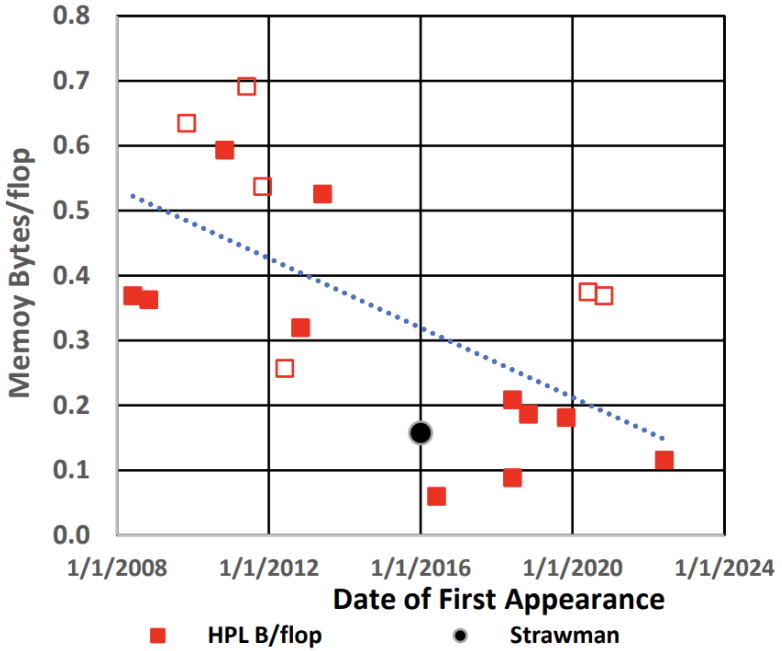
Georgia Tech

Exascale Data Movement Trends



Energy/FLOP decreasing roughly matches 2008 projections

Decreasing Byte/FLOP ratio decrease indicates our current systems are trending in the wrong direction!

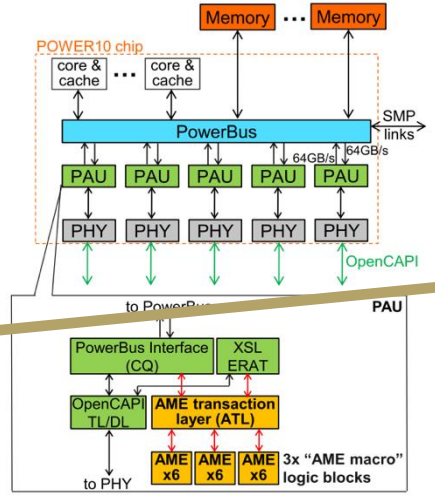


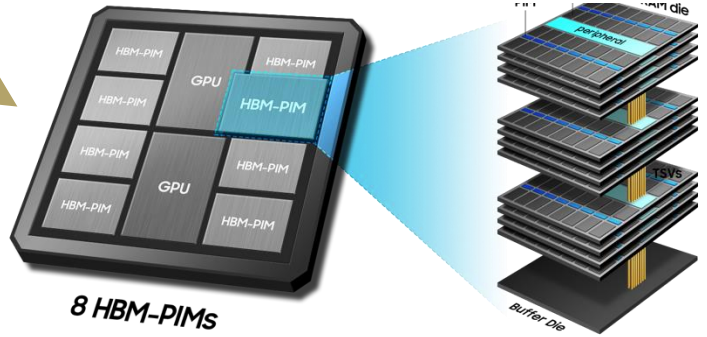
Clearly, we need to think differently about how we do data movement in the post-Moore and post-Exascale era!

P. M. Kogge and W. J. Dally, *Frontier vs the Exascale Report: Why so long? and Are We Really There Yet?*
PMBS 2022; doi: 10.1109/PMBS56514.2022.00008

Georgia Tech

Further Motivation

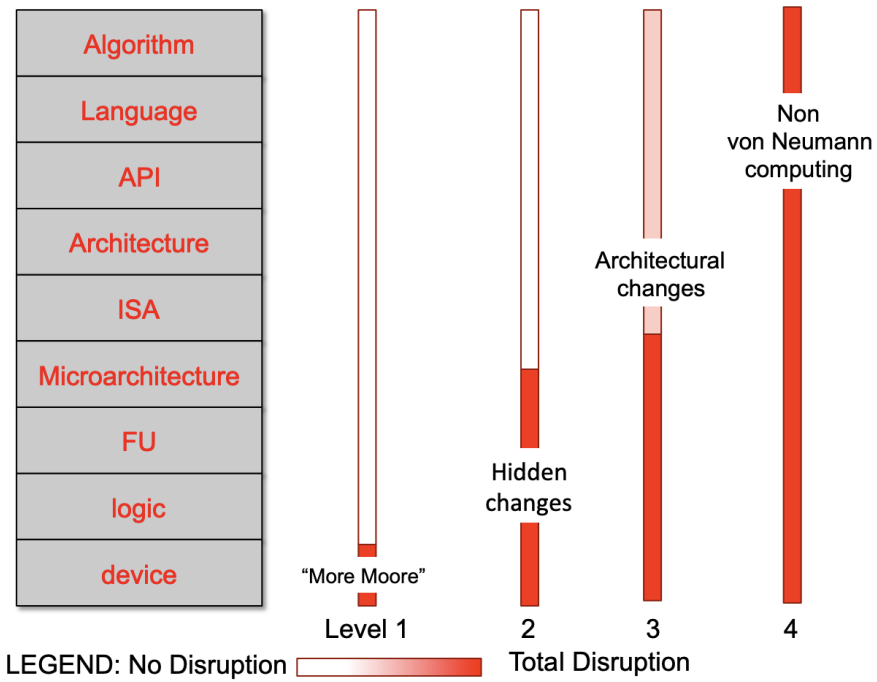




The "Cambrian Explosion" of new accelerators has led to many domain specific accelerators – not just in AI but also for data movement!

Georgia Tech

Center for Research into Novel Computing Hierarchies (CRNCH)



CRNCH was founded in 2017 to bring together researchers working all across the technology stack.

- *Both evolutionary and revolutionary improvements are needed to achieve performance in the post-Moore era*

The CRNCH Rogues Gallery testbed was created as a central point of focus for novel prototypes

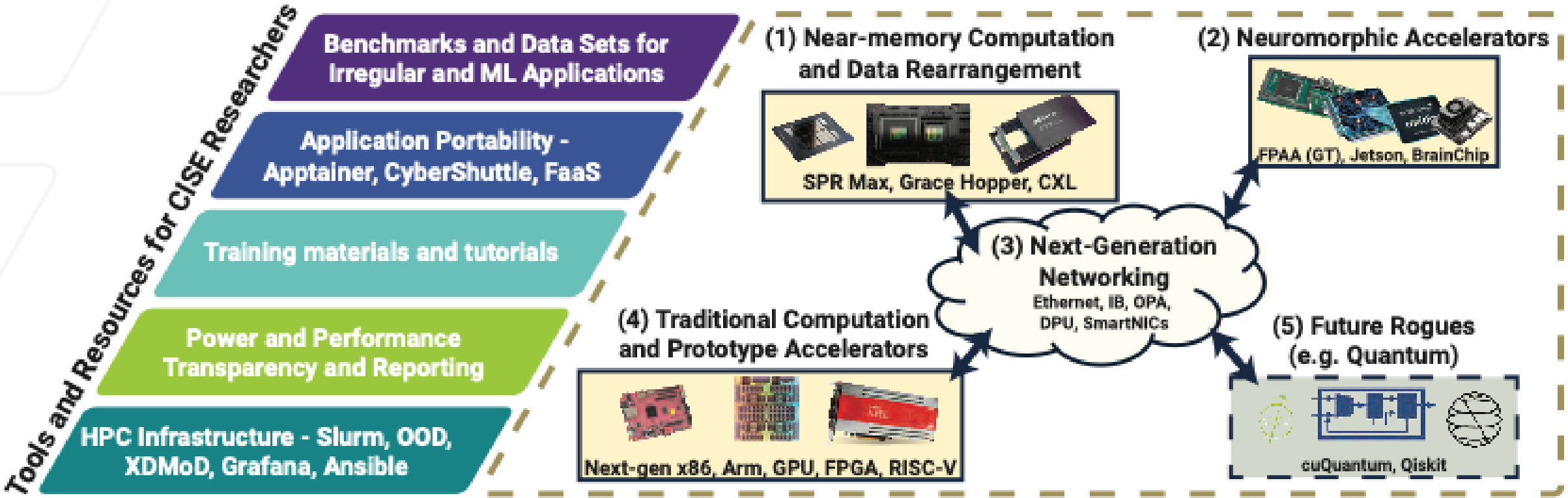
The Rogues Gallery

Rogues Gallery is an NSF funded post-Moore testbed for CISE researchers and the community.

- The Gallery contains 40 servers and 40+ development boards – Intel CLX, SKL, ICX; AMD/NVIDIA GPUs; Arm; RISC-V; Xilinx
- ***Extreme heterogeneity*** with GPU, FPGAs, FPAAs, Optane Memory, InfiniBand, OmniPath, and Ethernet networking
- This grant focuses on ***community engagement and post-Moore training***

Georgia Tech

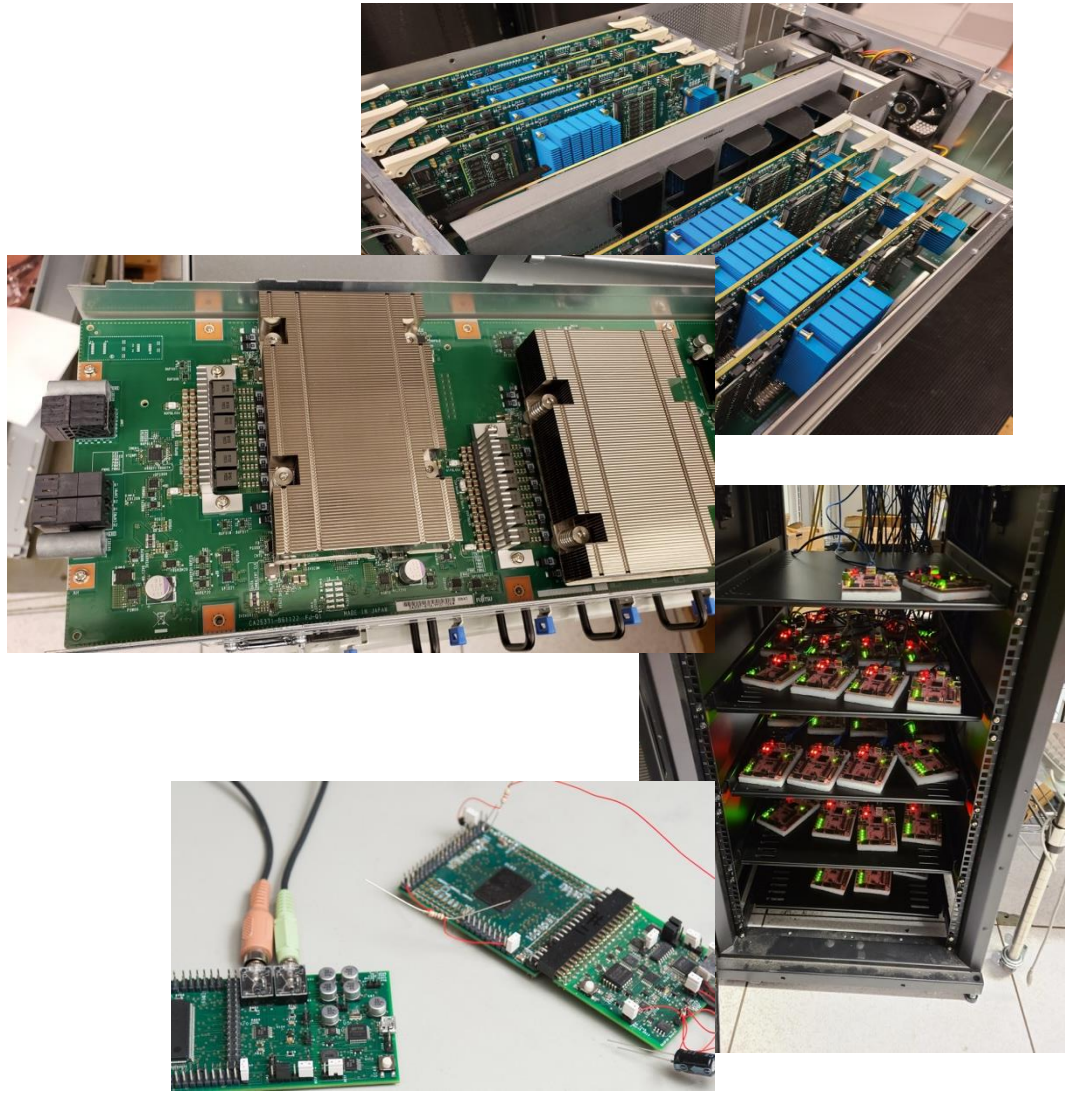
The Rogues Gallery



Georgia Tech

Rogues Gallery Highlights

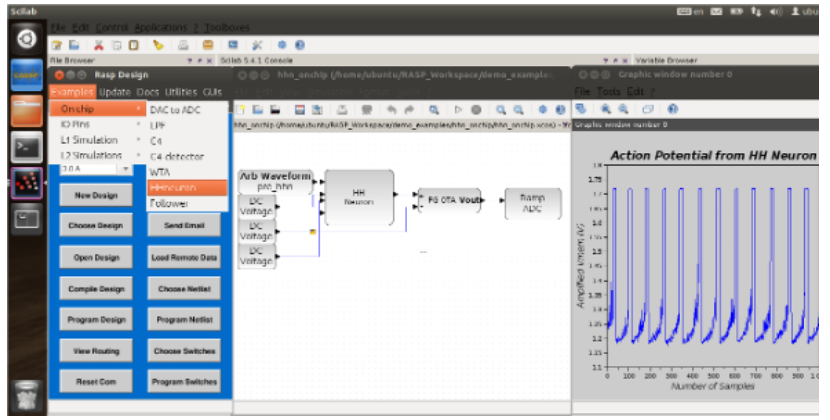
- First deployment of A64FX with Open OnDemand, later adopted by RIKEN for Fugaku
- Largest public instance of Lucata Pathfinder - #211 on 2021 Graph500 and #46 on GreenGraph500 rankings
- Support for over 180 researchers and 60-70 external users across multiple areas
- Support for 80-130 students each year with Pynq boards and similar infrastructure



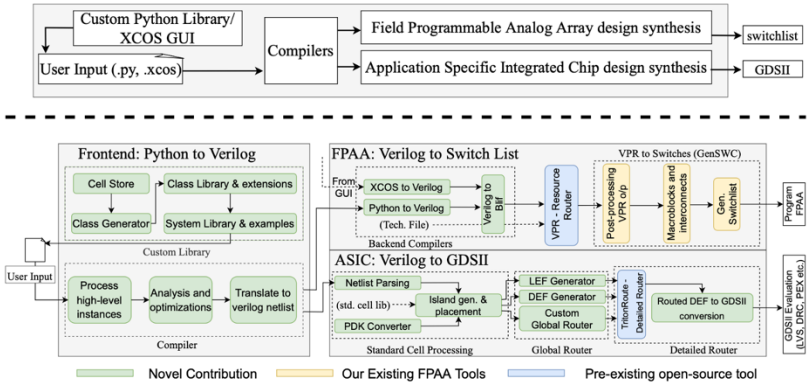
Georgia Tech

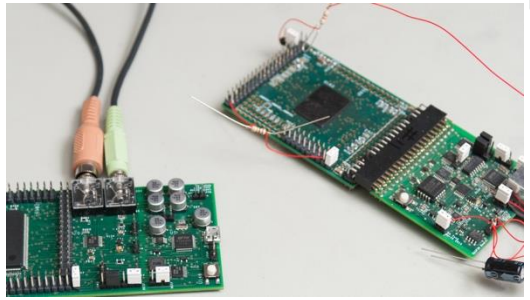
Unique Rogues Gallery Capabilities

- Support for investigating graph analytics accelerators, neuromorphic hardware, and smart networking
- Integration with local educational mission
  - Provides valuable testing of novel architecture
  - Ex: FPAA Python workflow, PYNQ cluster
- Growing interactions with software development best practices
  - CI/CD, continuous benchmarking



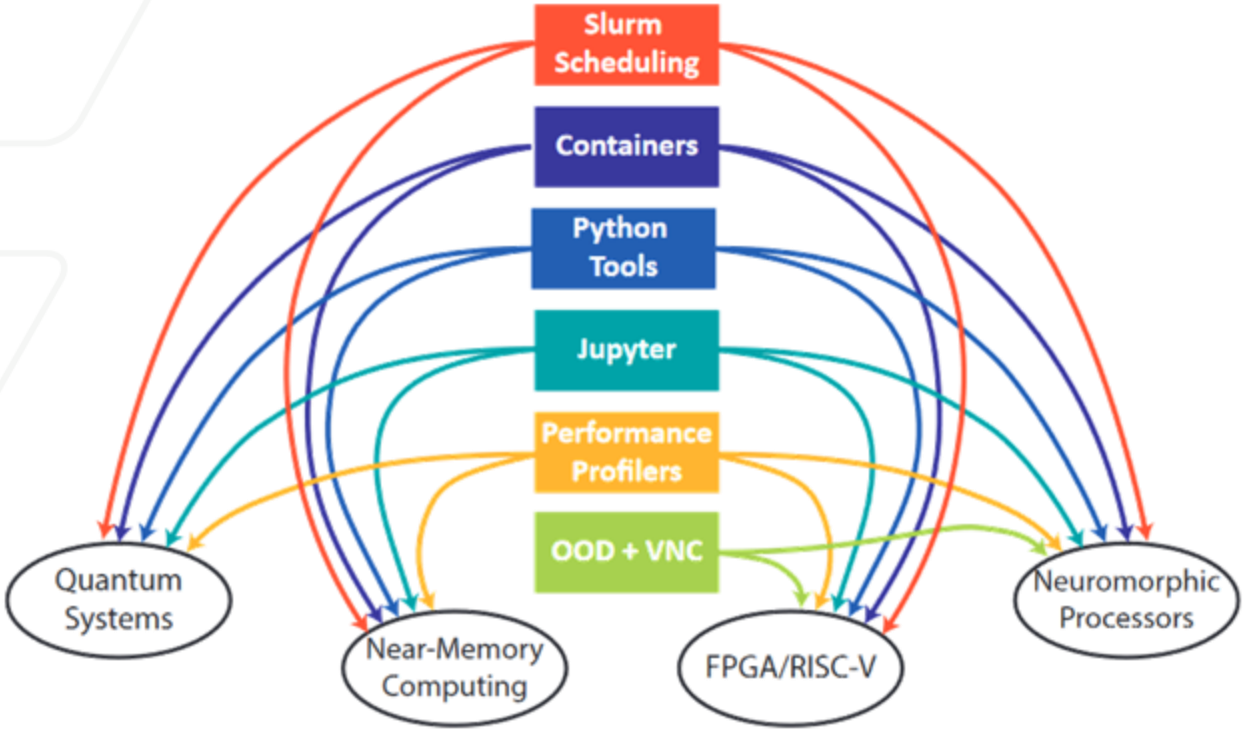
Next generation tools for FPAA development: graphical with Xcos interface (above), full open-source analog FPAA workflow (below)

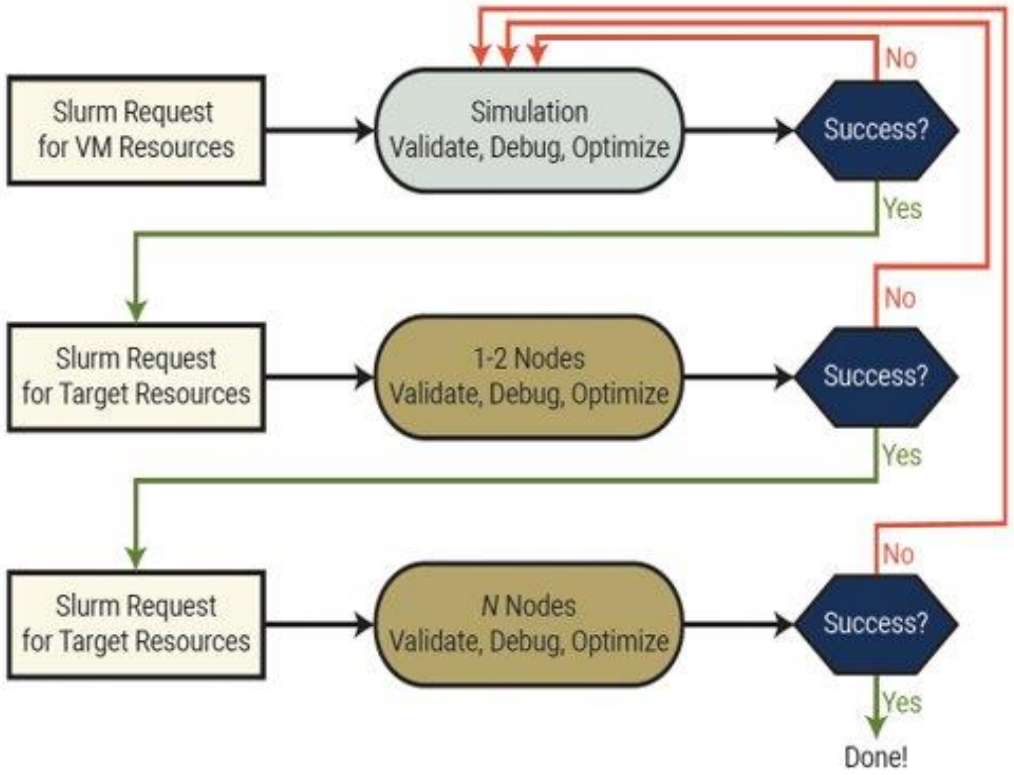




Georgia Tech

Managing Workflows for Novel Architectures

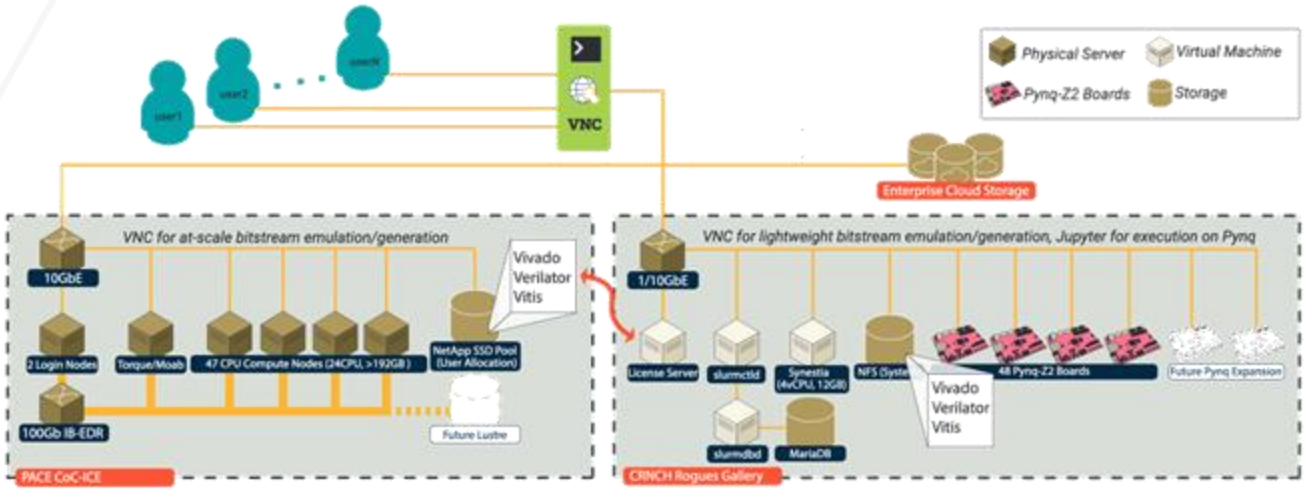




Georgia Tech

Slurm and Open OnDemand

- Slurm scheduling across most resources
  - Interesting challenges with “edge device” integration
  - Running 24.11 with 16 different builds (x86, Arm, NVIDIA, RISC-V, etc.)
- Open OnDemand support to ease onboarding



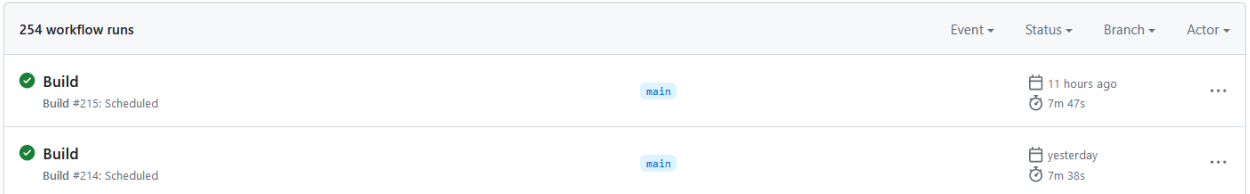
Jezghani A, Manalo K, Powell W, Valdez J, Young, J. *Onboarding Users to A64FX via Open OnDemand*. International Conference on High Performance Computing in Asia-Pacific Region Workshops. 2022; 78-83. https://doi.org/10.1145/3503470.3503479

Georgia Tech

CI/CD support on the Rogues Gallery

- Part of supporting novel architectures includes supporting *effective software development*
  - CRNCH RG provides Slurm-oriented infrastructure to schedule runners for novel architectures (RISC-V, Arm, Intel GPU, FPGA)
    - Dedicated VM and secure setup process allows for persistent runners

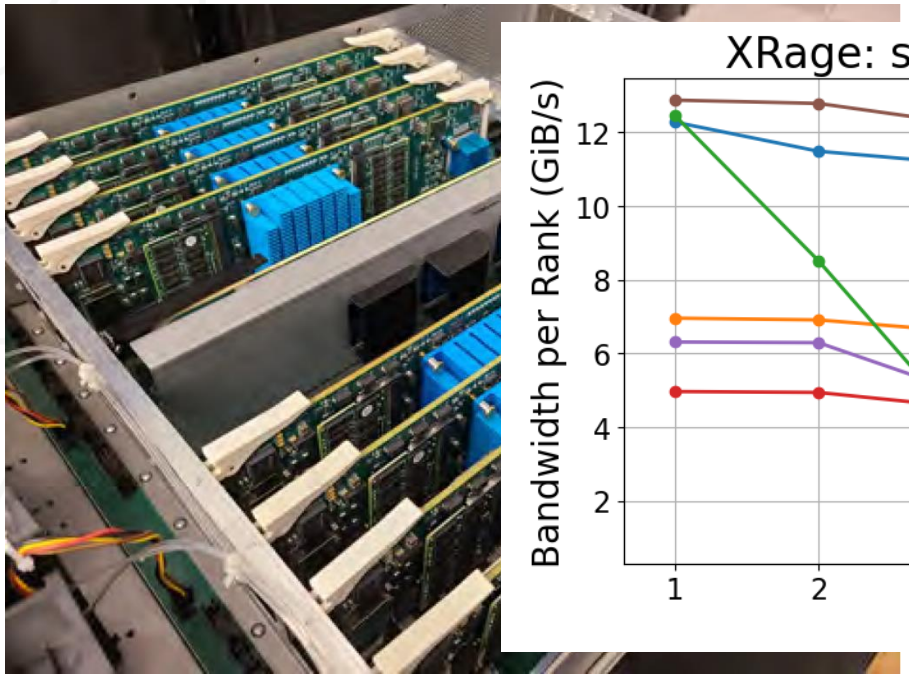

```
# Authentication

√ Connected to GitHub

# Runner Registration

Enter the name of the runner group to add this runner to: [press Enter for Default]

Enter the name of runner: [press Enter for rg-ci-workflow1] spatter-nvidia-gpu

This runner will have the following labels: 'self-hosted', 'Linux', 'X64'
Enter any additional labels (ex. label-1,label-2): [press Enter to skip]

√ Runner successfully added
√ Runner connection is good

# Runner settings

Enter name of work folder: [press Enter for _work]

√ Settings Saved.
```

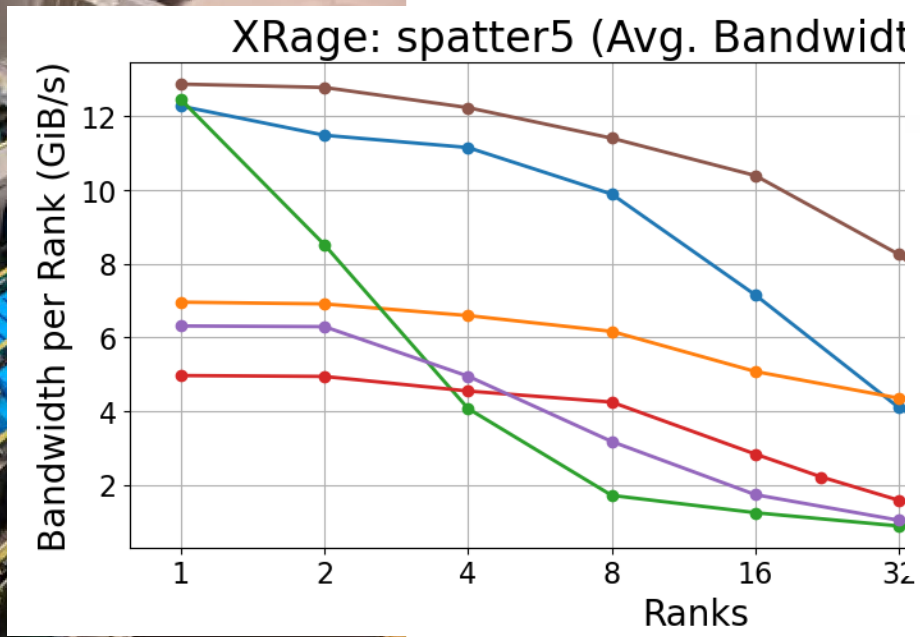


Learn More: https://gt-crnch-rg.readthedocs.io/en/main/general/ci-runners.html

Georgia Tech

Case Studies of How We Use the RG Testbed





Lucata Pathfinder – Move the Compute to the Data

SmartNICs – What are they Good For?

Spatter - Simulating Next-Generation Memory Systems

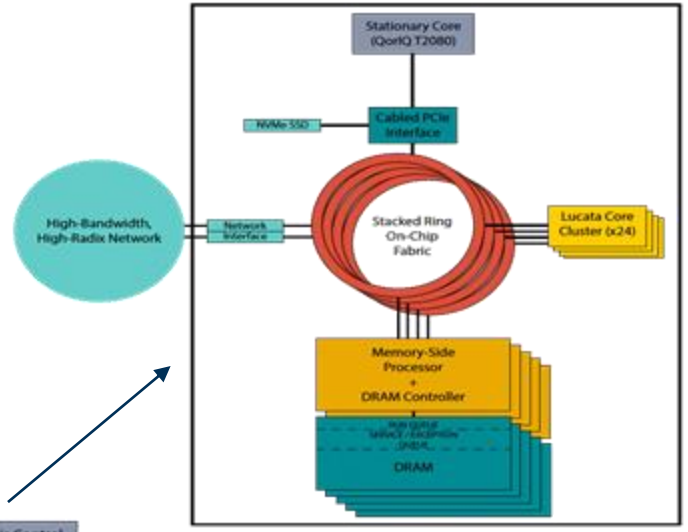
Georgia Tech

Case Study 1 – Move the Compute to the Data

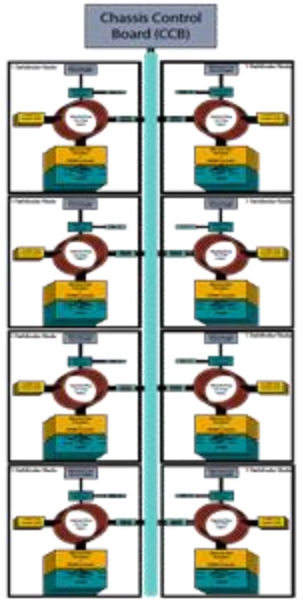
- The Lucata Pathfinder is the next iteration of Emu Chick prototype
- Design motivated by scaling issues for problems hindered by data access patterns
  - E.g. graphs, sparse matrix algebra
- Rather than migrating large swaths of data across network, hardware threads can move to the memory
  - Demonstrated scaling efficiency with benchmarks such as BFS, hpcg vs. x86

Pathfinder - Reconfigurable Hardware

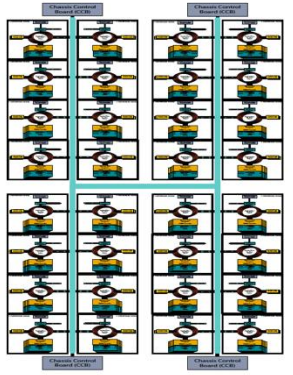
- Each chassis contains:
  - 1 Chassis Control Board (PowerPC E6500) for node/network management
  - 8 compute nodes, which include:
    - 1 stationary compute core (PowerPC E6500)
    - 24 Lucata compute cores (Proprietary IP)
    - Stacked ring-on-chip fabric to interface to network/DRAM/compute
    - High-bandwidth, high-radix network for inter-node/chassis communication
- Systems can be configured as single or multi-chassis
  - Defines network and available Lucata compute cores for running applications
  - Scale hardware according to the problem size
- GT Pathfinder consists of 4 chassis



1 Pathfinder Node



8 Nodes per Chassis


4 Nodes in GT Pathfinder

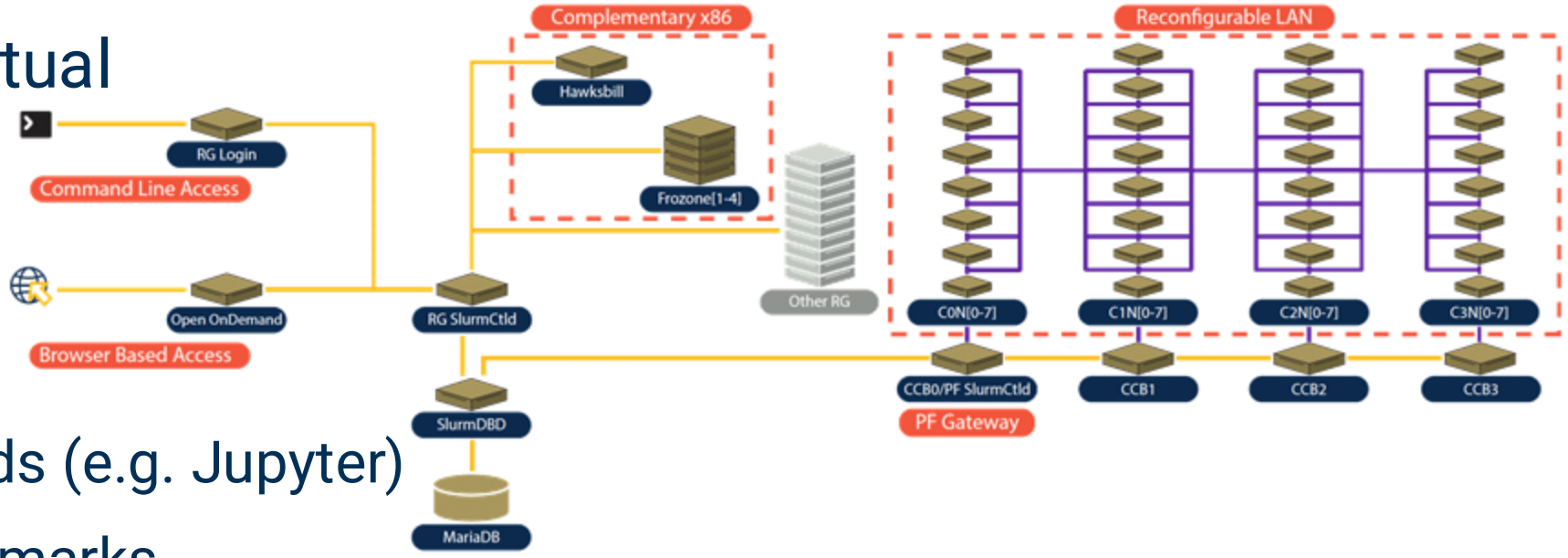
Georgia Tech

High-Level Libraries/APIs for Code

- Lucata-provided software environment
  - Custom adaptation of Yocto Linux kernel and base utilities
  - Configuration tools to dynamically modify chassis, network
  - Performance monitoring libraries to check efficiency, energy
  - Custom API for thread management and task execution
  - Robust simulation capabilities to validate before deployment
- Open-Cilk for parallel programming
  - Open-source platform for task-parallel code development
  - Leverages lightweight backend ABIs for target dispatch
  - Minimal effort to port code from x86 to Pathfinder

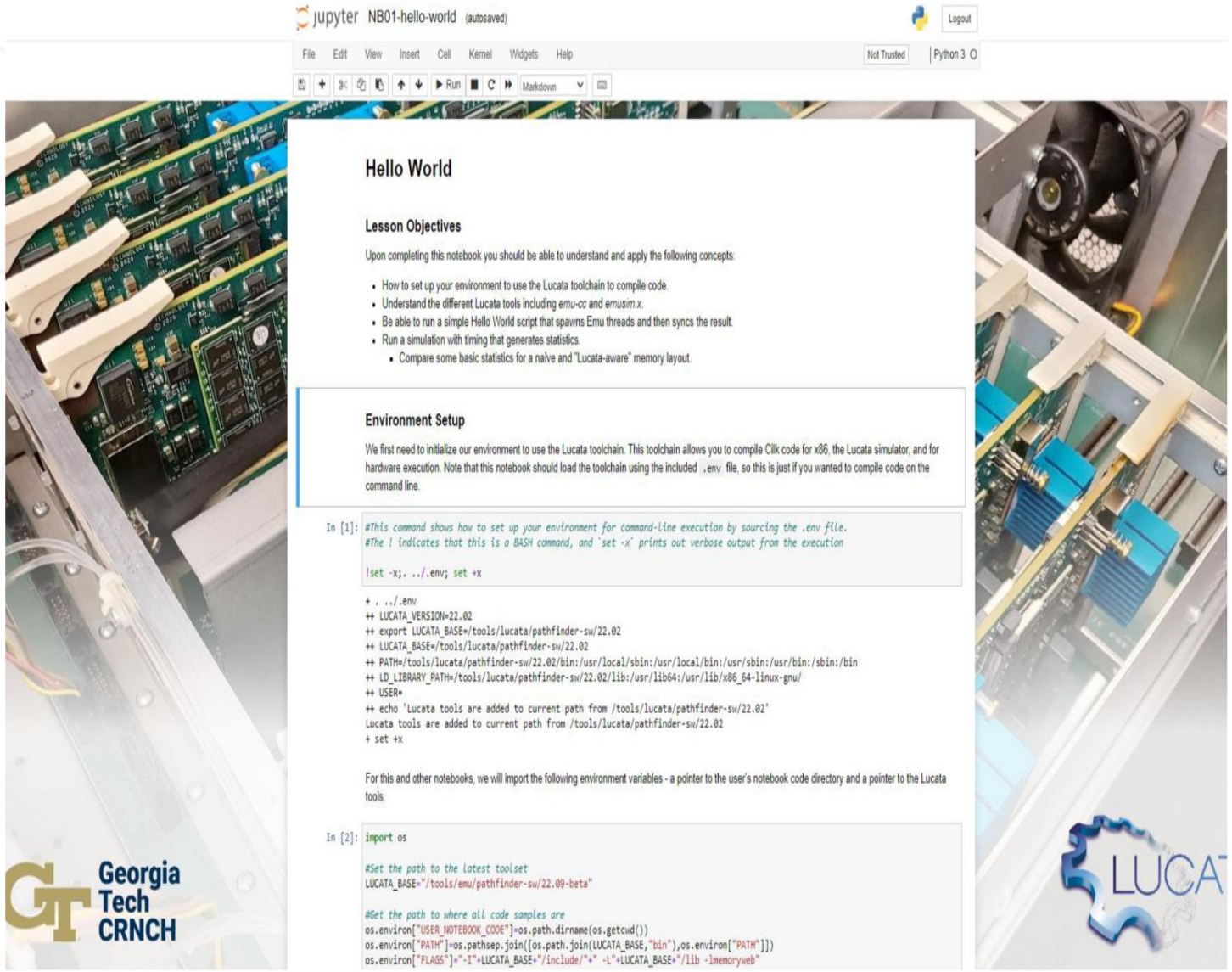
Accessing the Pathfinder and Related Systems

- Pathfinder access managed via Slurm (and calendar schedule)
  - Federated cluster alongside other Rogues and infrastructure nodes
  - Single entry point, single job database
- x86 physical and virtual machines for development and simulation
  - Host visual frontends (e.g. Jupyter)
  - Comparative benchmarks



Georgia Tech

Community Tutorials to Engage Researchers



Tutorials covering Emu Chick/Lucata Pathfinder presented at numerous conferences

- General RG overview at ASPLOS19 and PEARC19
- Dedicated tutorial at PEARC21 and HPEC22
- Iteratively developed content and workflows, such as Jupyter-based approach for HPEC22

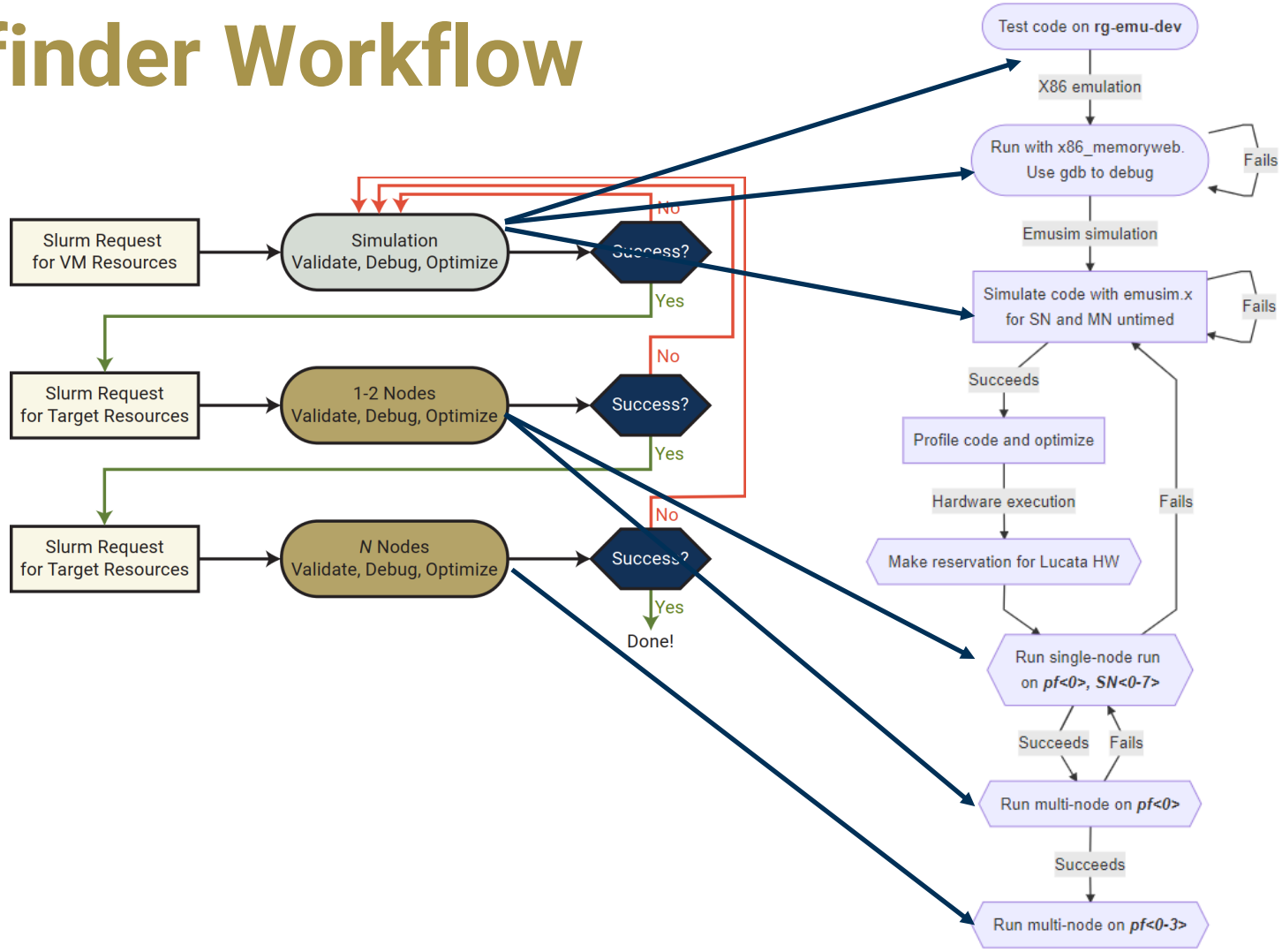
Lessons learned for the platform and tutorials in general:

- User engagement depends on user interface and prerequisite knowledge
- Conferences with fewer cross-listed and hybrid options increased attendance and participation
- Attendees may have HPC experience, but still need more focused examples for new architectures

Georgia Tech

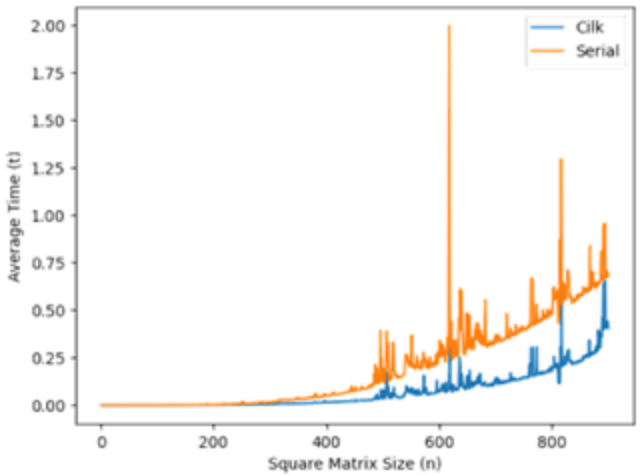
Full Lucata Pathfinder Workflow

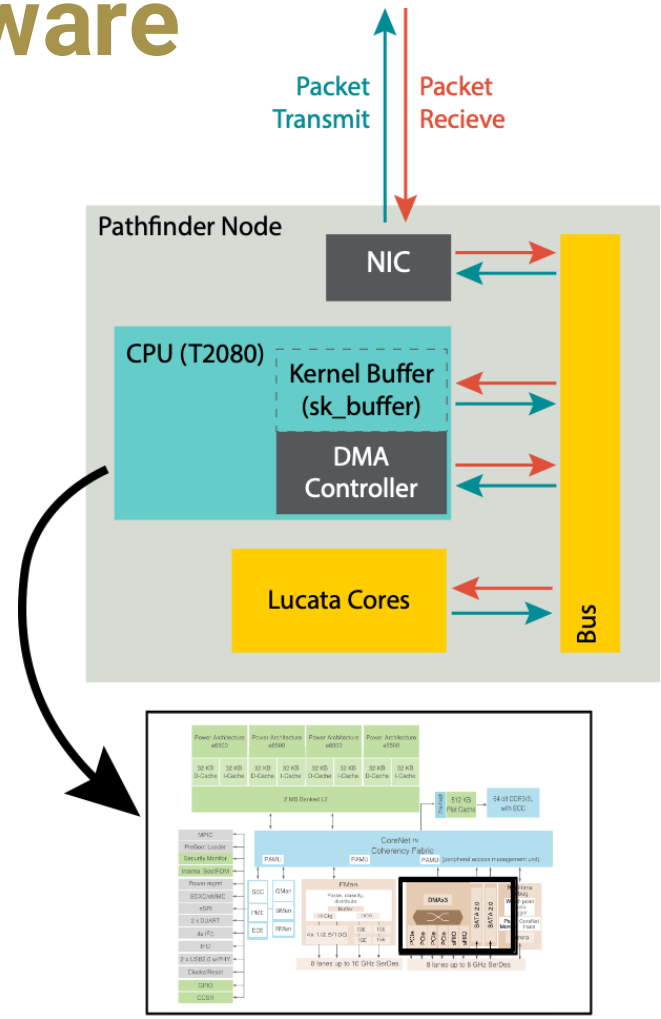
Working with the Lucata Pathfinder still requires a complex workflow, but we can break it down into small, concrete steps for users



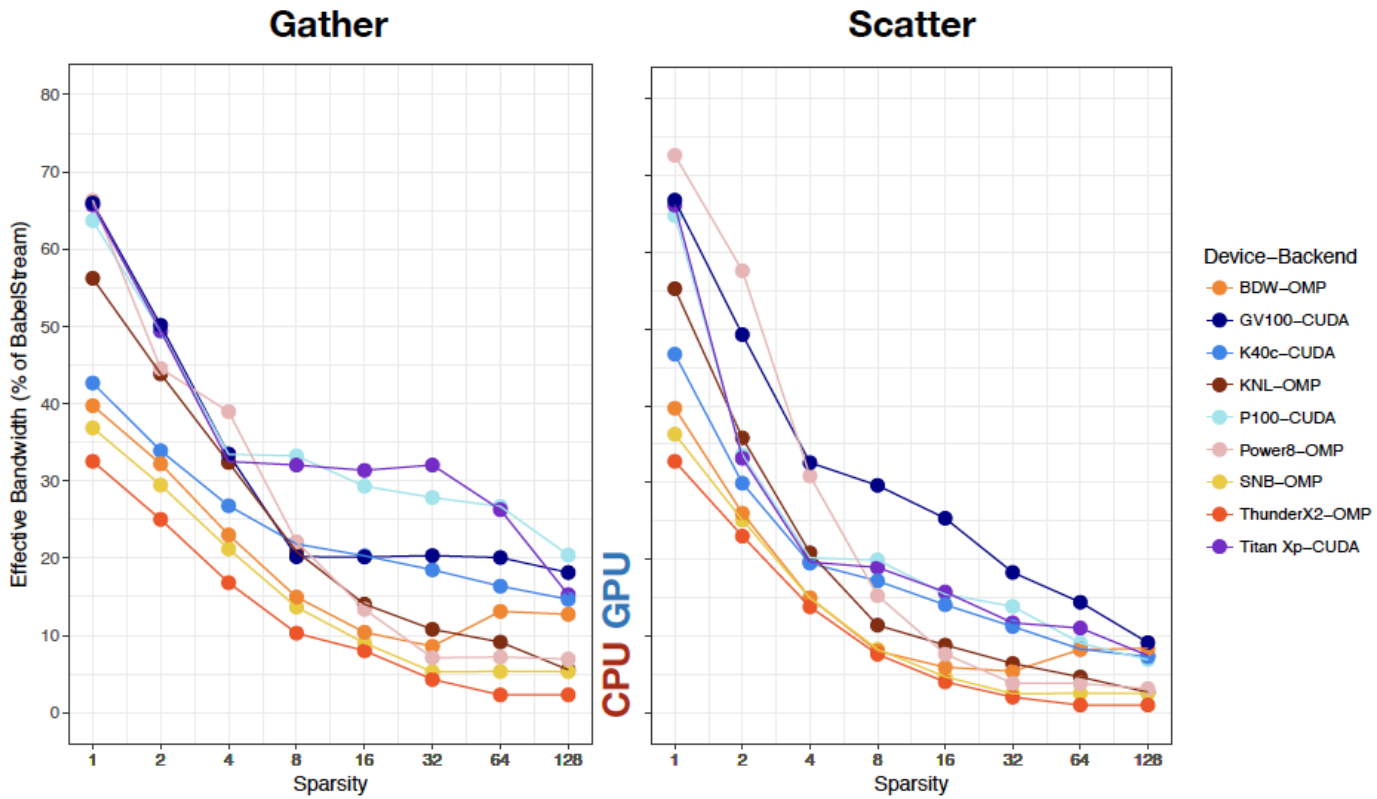
Extending CS Curriculum to Novel Hardware

- Future Computing with the Rogues Gallery VIP
  - Near-memory subteam focuses on the Pathfinder architecture for parallel and distributed computing
- Explore benchmarks on x86 and Pathfinder systems
  - hpcg to assess scaling performance for SpMV
  - BFS to demonstrate graph capabilities
  - Pointer Chase (pChase) to compare memory latency
- DMA driver prototype for Lucata compute cores
  - Project option for CS3210: Design of Operating Systems





Case Study 2 – Evaluating Next-generation Architecture Designs for Memory Systems



See Kevin Sheridan, et. al, "A Workflow for the Synthesis of Irregular Memory Access Microbenchmarks." In *Proceedings of the International Symposium on Memory Systems*, pp. 219-234. 2024.

Georgia Tech

Motivation

	Memory subsystem						Floating Point		
	L1	L2	L3	DRAM	DRAM BW	Mem Latency	DP FLOPs	Vectorization	Non-FP
Flag 3D Ale							2.50%	7.10%	97.50%
PartiSN 42 groups							26.20%	90.40%	73.80%
Jayenne DDMC Hohlraum							14.30%	0.20%	85.70%
xRAGE Shaped Charge							6.50%	14.00%	93.50%
Application 1							7.80%	19.20%	92.20%
Application 2							8.10%	17.60%	91.90%

Hardware Bottlenecks for LANL HPC codes [1]

However, we still have little understanding of how future memory accelerators might affect codes of interest because:

- Finding appropriate regions of interest (ROI) for the memory system is challenging to infer even with tools like SimPoints and LoopPoint
- Some applications that we might want to benchmark can't be fully shared to extract meaningful traces or ROIs

Our ideal workflow would allow us to 1) capture relevant memory accesses from real-world applications, 2) benchmark them on real systems, and 3) simulate new hardware models

[1] G. Shipman, et al., Assessing the Memory Wall in Complex Codes, MCHPC 2022 doi: 10.1109/MCHPC56545.2022.00009

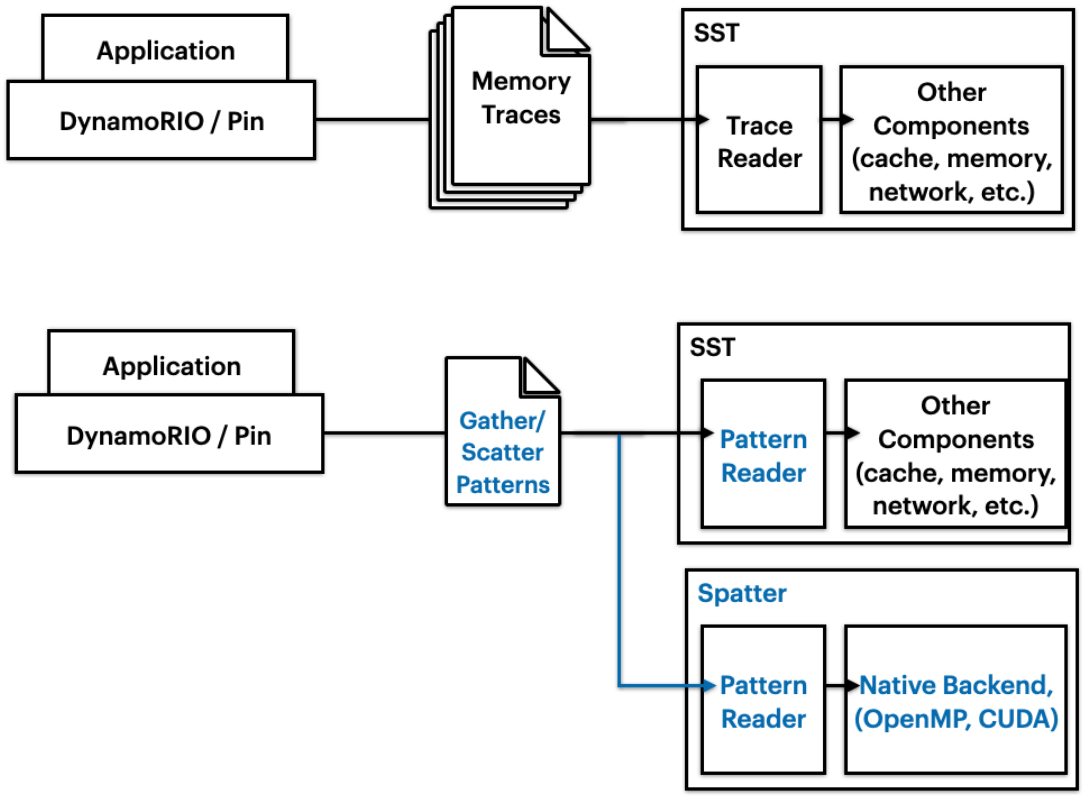
Using Codesign to Design Better Memory Systems

Our ideal tool workflow would allow us to:

- Capture relevant memory accesses from real-world applications

- Run these patterns on bleeding edge systems

- Use the ***same patterns*** to simulate the performance of future near-memory accelerators



Georgia Tech

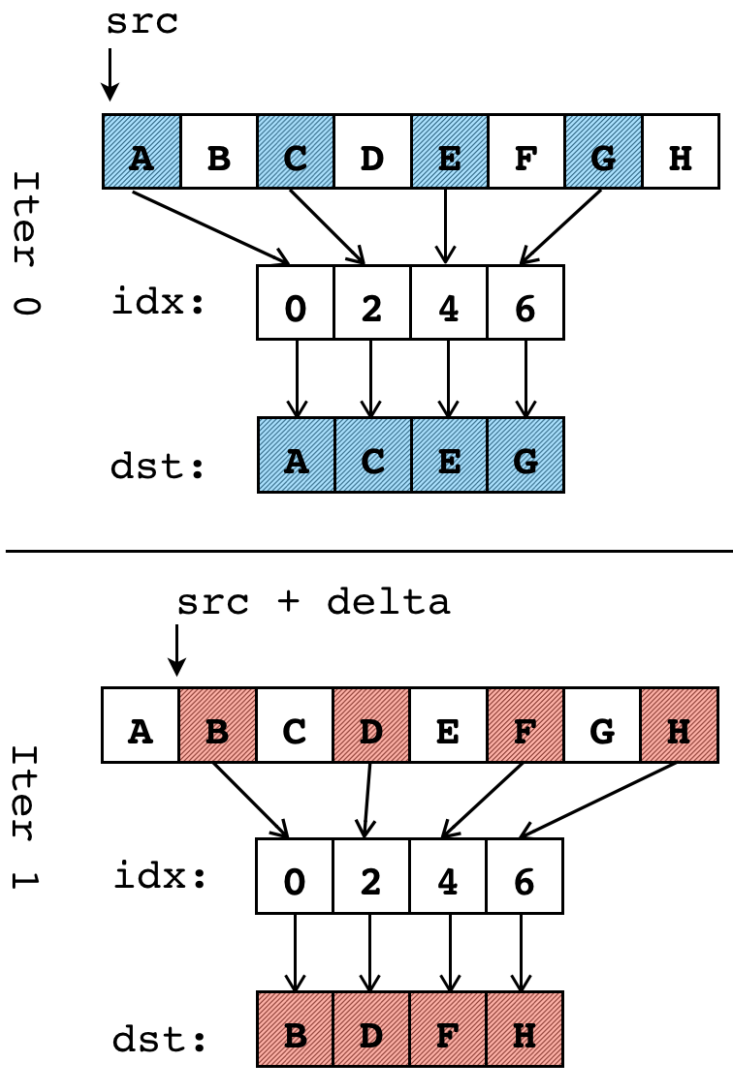
Spatter - Version 1.0

The basis of Spatter is two kernels; one for gather and another for scatter.

Gather kernel:
for i in 0..N:
    reg = gather(src + delta*i, idx)

Scatter kernel:
for i in 0..N:
    scatter(dst + delta*i, idx, reg)

The delta and the pattern in idx specify the *memory access pattern*.



Georgia Tech

GS Patterns

Sliding window approach is used to track non-trivial memory accesses within an application trace or region of interest (ROI)

$$\textbf{ScatterWindow} = \begin{array}{c} \begin{array}{cccc} \text{maddr1} & \text{maddr2} & \ldots & \text{maddrN} \end{array} \\ \begin{bmatrix} 0x0480 & 0x0488 & \ldots & 0x1488 \\ 0x1780 & 0x1788 & \ldots & 0x0783 \\ \ldots & \ldots & \ldots & \ldots \\ 0x9580 & 0x5588 & \ldots & 0x3581 \end{bmatrix} \end{array} \begin{array}{c} iaddr \\ 0x0001 \\ 0x0003 \\ \ldots \\ 0x1019 \end{array}$$

Multiple filters are used to keep the most relevant access patterns for the final pattern output

Filter #	Description
1	Index distances are only -1, 0, and/or 1
2	No symbol
3	Not in top 10 window appearance counts
4	Less than 1024 instances
5	Less than 6 unique index distances and less than 50% out of bounds distances*

*Filter rules apply to each full memory access sequence. Sub-sequences are not removed. *Default out of bounds distances in $(-\infty, -513]$ or $[513, \infty)$.*

GS Patterns codebase at https://github.com/lanl/gs_patterns

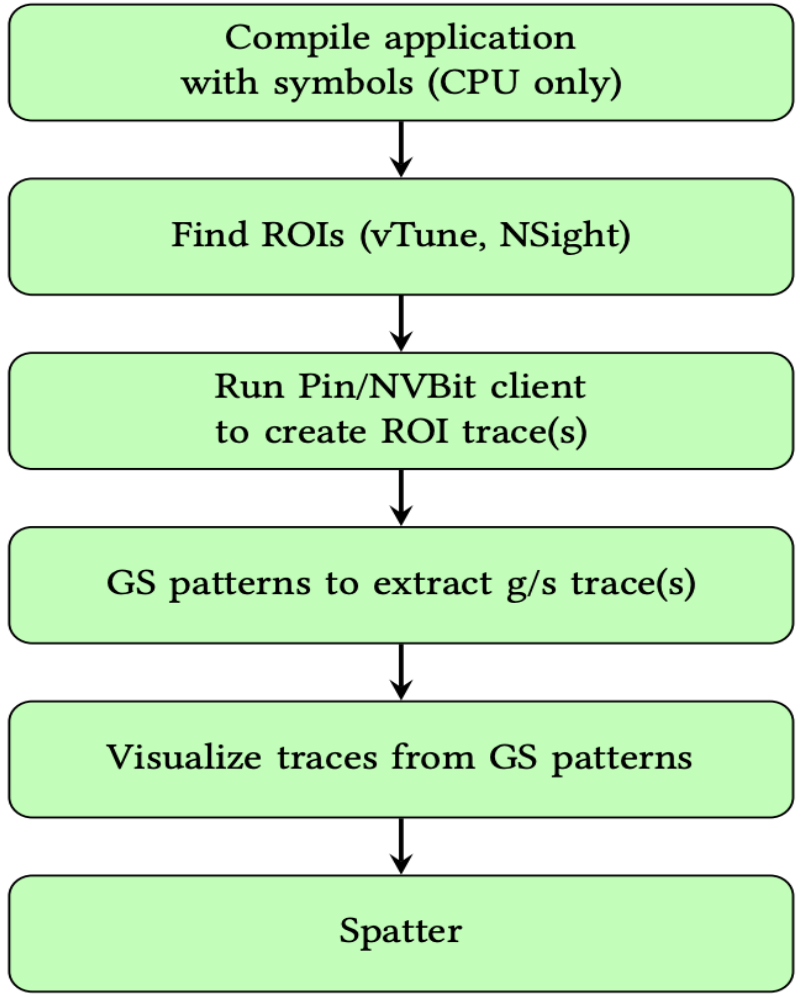
Georgia Tech

GS Patterns Workflow

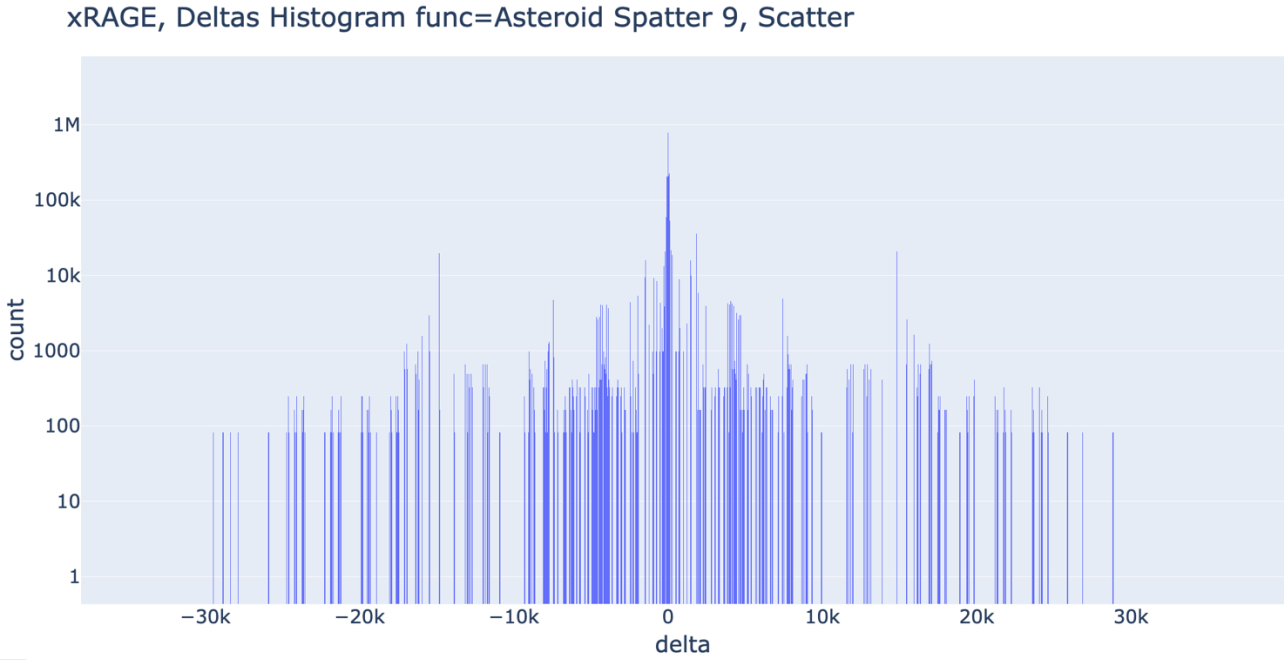
Further extensions to GS Patterns refactored code to use C++ and a plugin infrastructure for PinTool, NVBit

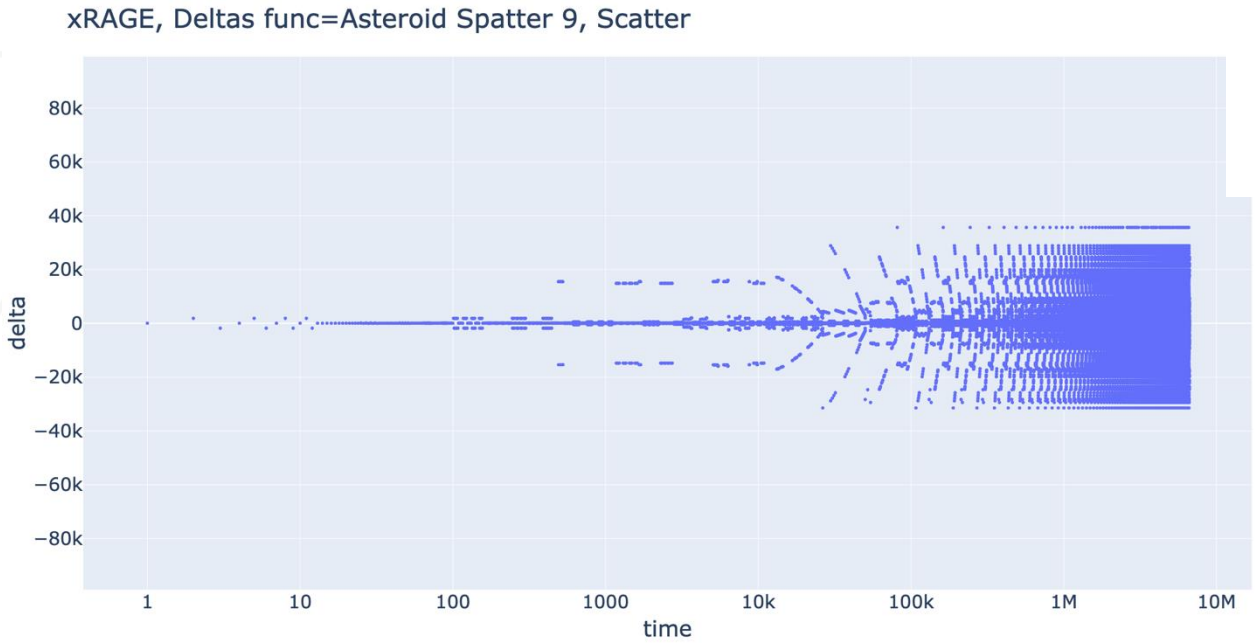
Currently ROI analysis speeds up the overall GS Patterns workflow but is a somewhat manual process to run and annotate codes

Check out collected public patterns at https://github.com/hpcgarage/spatter-patterns



Pattern Visualization Tools



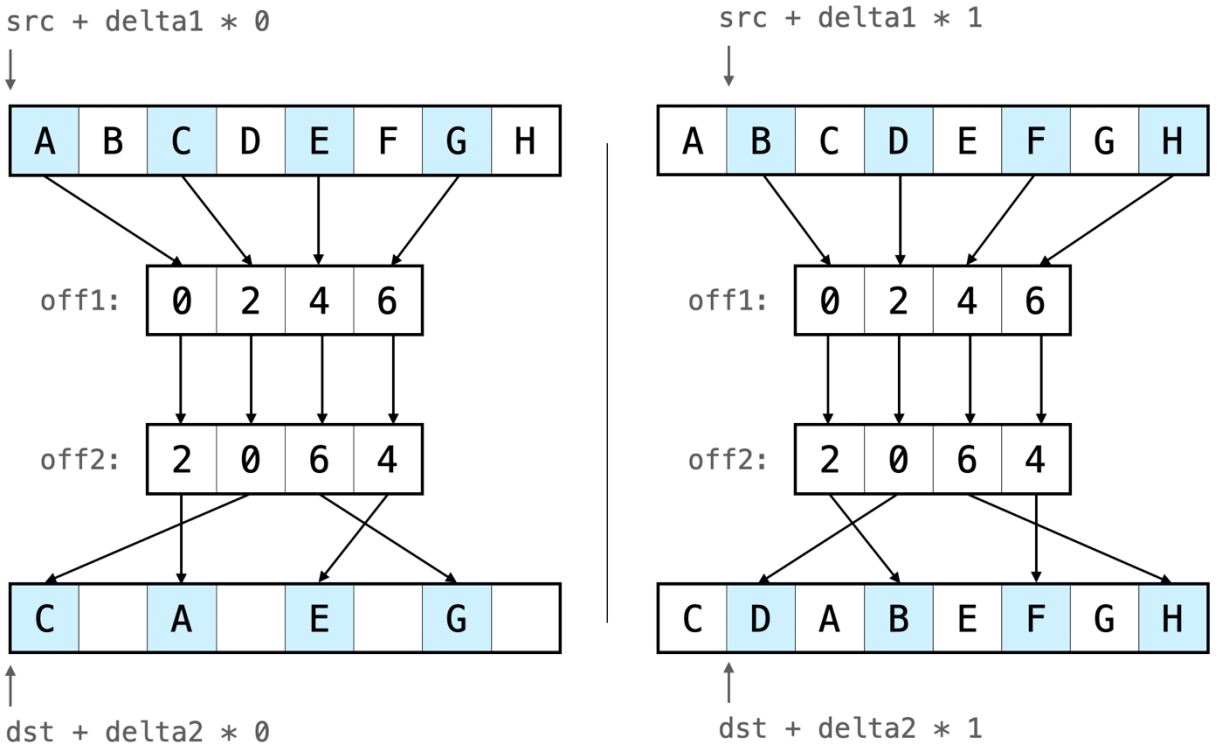


The GS Patterns JSON output can be easily visualized in a variety of different ways…

Georgia Tech

Spatter 2.0

- Complete refactor of argument parsing, build system, and movement towards C++ design
- Addition of new kernels to better represent multiple levels of indirection - GatherScatter, MultiGather, MultiScatter
- Support for longer offset buffer lengths for improved application pattern representation
- MPI support for weak/strong scaling
- Support for atomics with scatter operations



GatherScatter Kernel Representation

https://github.com/hpcgarage/spatter

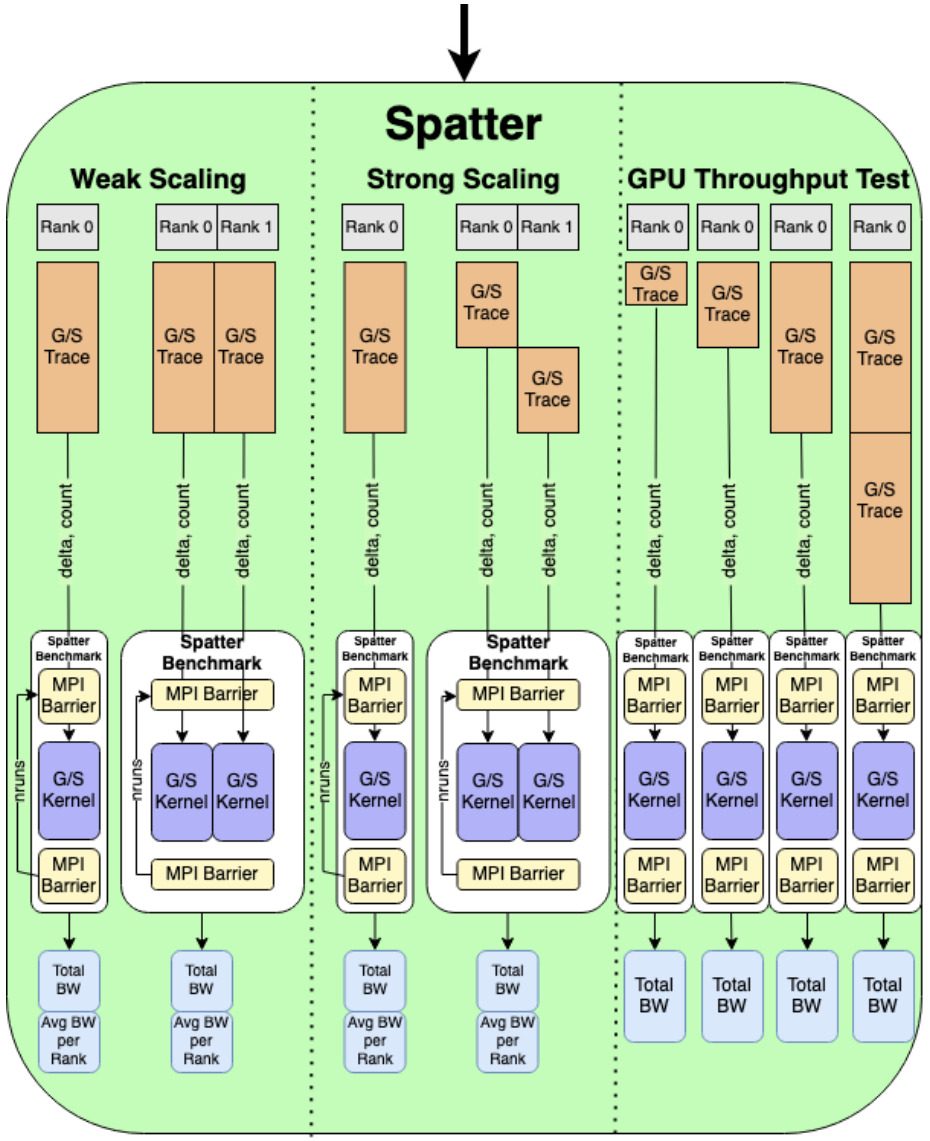
Georgia Tech

Spatter 2.0 MPI Workflow

Weak Scaling – each MPI rank gets the same pattern and scaling scripts are used to sweep across N ranks

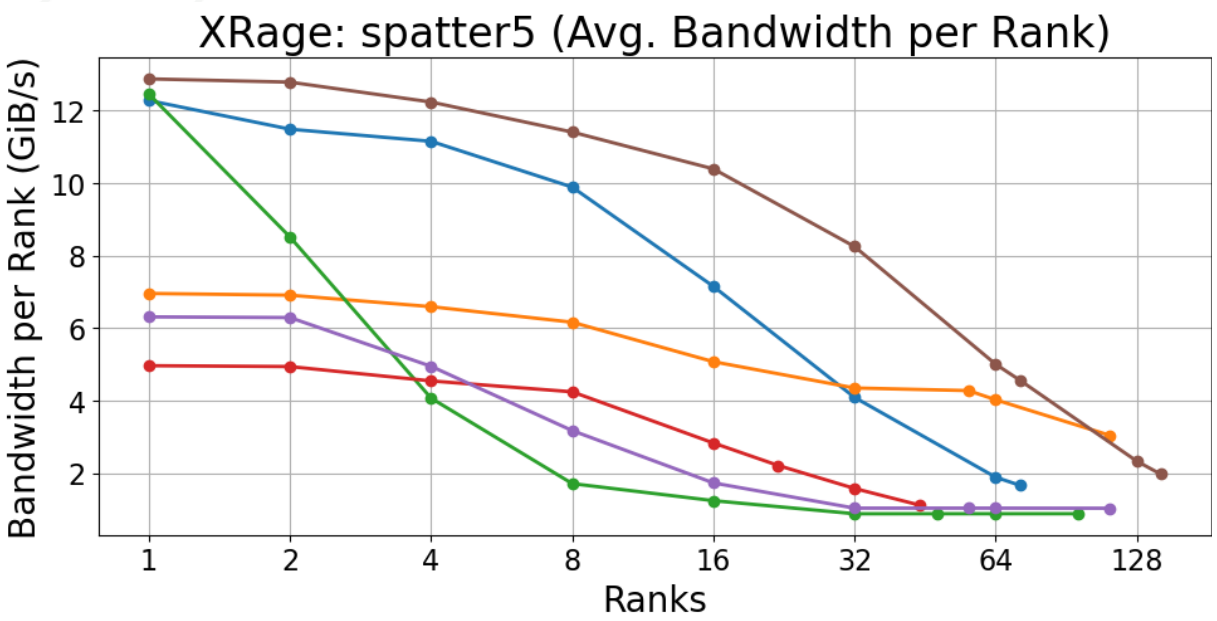
Strong Scaling – access patterns are partitioned across MPI ranks

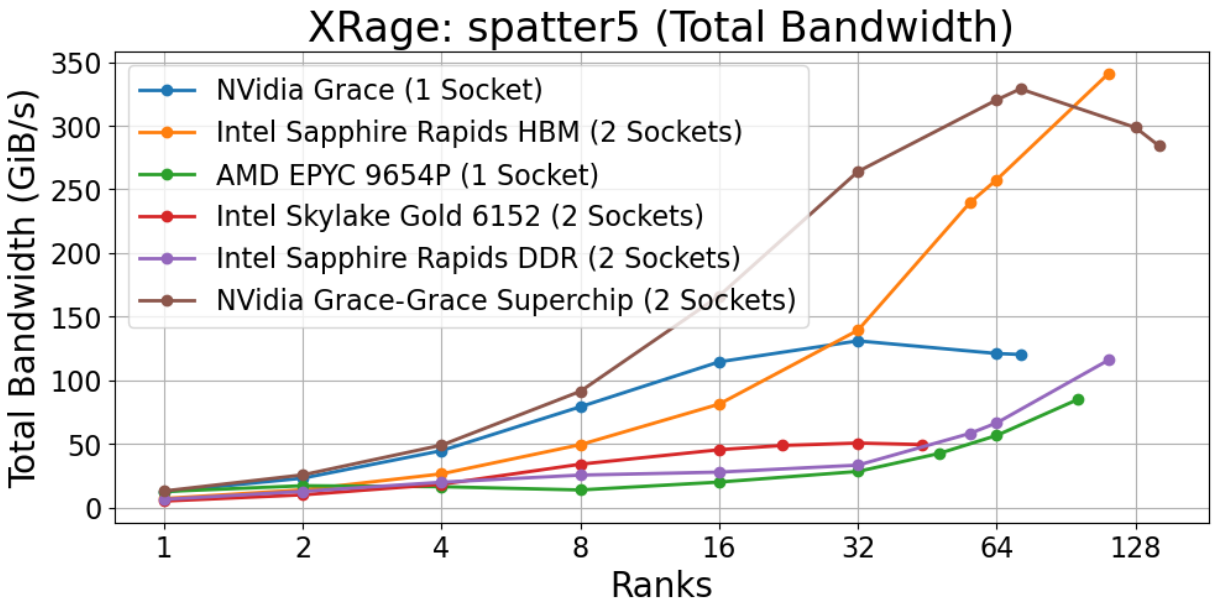
GPU Throughput – patterns are truncated or expanded to vary amount of memory accesses and saturate GPU memory subsystem



Georgia Tech

Spatter 2.0 Results





Weak Scaling MPI Tests for Xrage Gather pattern

For more details, please see: Kevin Sheridan, et. al, "A Workflow for the Synthesis of Irregular Memory Access Microbenchmarks." In Proceedings of the International Symposium on Memory Systems, pp. 219-234. 2024.

Georgia Tech

Spatter Integration with SST

```
Running Spatter version 1.1
Compiler: GNU ver. 11.4.0
Backend: Serial
Aggregate Results? NO

Run Configurations
[ {'id': 0, 'kernel': 'scatter', 'pattern': [0, 24, 48, 72, 96, 120, 144
    'pattern-gather': [], 'pattern-scatter': [], 'delta': 0, 'delta-gath
   ....
  {'id': 11, 'kernel': 'gather', 'pattern': [0, 24, 48, 72, 96, 120, 144
    'pattern-gather': [], 'pattern-scatter': [], 'delta': 1, 'delta-gath

config          bytes           time(s)         bw(MB/s)
0               73959168        0.0025168       29386.1
1               29593344        0.00101654      29111.8
2               21479040        0.000900899     23841.8
3               16384256        0.00103612      15813.1
4               12334080        0.000430775     28632.3
5               12334080        0.000427073     28880.5
6               12311808        0.000426331     28878.5
7               11265408        0.000443739     25387.5
8               9829632         0.000339094     28987.9
9               9829632         0.000783912     12539.2
10              9829632         0.000334772     29362.2
11              9250560         0.000322094     28720.1
```

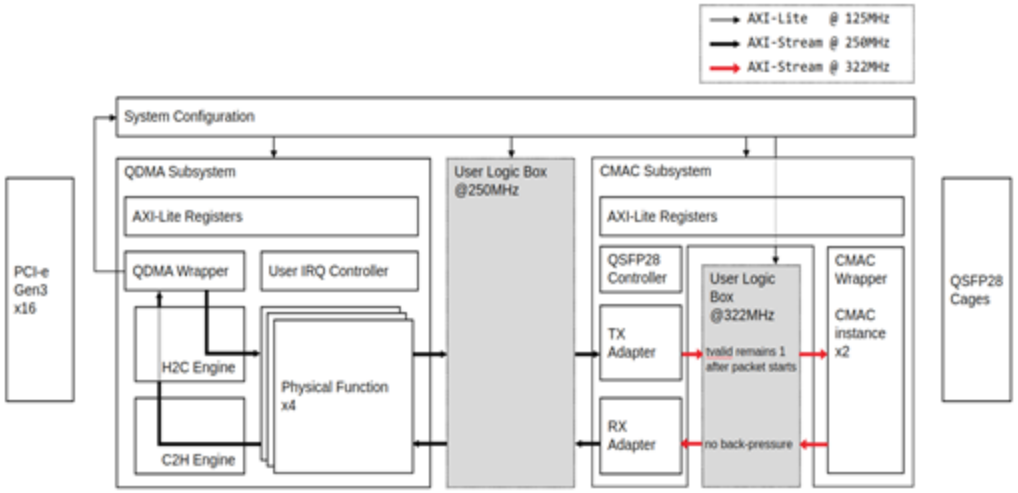
```
Run Configurations
[ {'id': 0, 'kernel': 'scatter', 'pattern': [0, 24, 48, 72, 96, 120, 144, 168, 192, 216, 240, 264, 288, 312, 336, 360],
      'pattern-gather': [], 'pattern-scatter': [], 'delta': 0, 'delta-gather': 8, 'delta-scatter': 8, 'count': 577806, 'wrap': 1, 'threads': 1},

  {'id': 11, 'kernel': 'gather', 'pattern': [0, 24, 48, 72, 96, 120, 144, 168, 192, 216, 240, 264, 288, 312, 336, 360],
      'pattern-gather': [], 'pattern-scatter': [], 'delta': 1, 'delta-gather': 8, 'delta-scatter': 8, 'count': 72270, 'wrap': 1, 'threads': 1} ]


config          bytes           time(s)         bw(MB/s)        cycles          time(s)/cycles
0               73959168        0.0104005       7111.11         5489157         1.89474e-09
1               29593344        0.00410376      7211.27         2300269         1.78404e-09
2               21479040        0.00297854      7211.26         1669566         1.78402e-09
3               16384256        0.00230403      7111.12         3393131         6.79029e-10
4               12334080        0.00163812      7529.40         1259456         1.30066e-09
5               12334080        0.00154176      8000.01         2530289         6.09321e-10
6               12311808        0.00153897      8000.01         2525826         6.09296e-10
7               11265408        0.0015622       7211.25         826265          1.89067e-09
8               9829632         0.00136309      7211.30         2024452         6.73312e-10
9               9829632         0.00136309      7211.28         10223443        1.3333e-10
10              9829632         0.00136308      7211.32         1212079         1.12458e-09
11              9250560         0.00128279      7211.26         678285          1.89123e-09

Simulation is complete, simulated time: 44.5121 ms
```

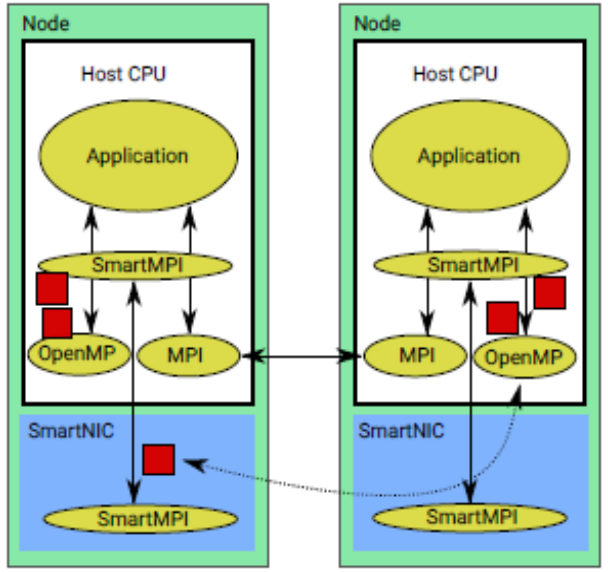
Lulesh Output (i5-12400F Alder Lake vs. SST Miranda Spatterbench)

Larger application patterns may benefit from either parallel simulation sweeps or by using MPI-based tests

Georgia Tech

Case Study 3 – SmartNICs for HPC?





Example of task-based runtime work by *Weinzierl, et al. "Intelligent algorithms on intelligent networks—experiences and challenges using NVIDIA's BlueField technology". Slides at https://dpu.ornl.gov/*

Georgia Tech

What's a SmartNIC?

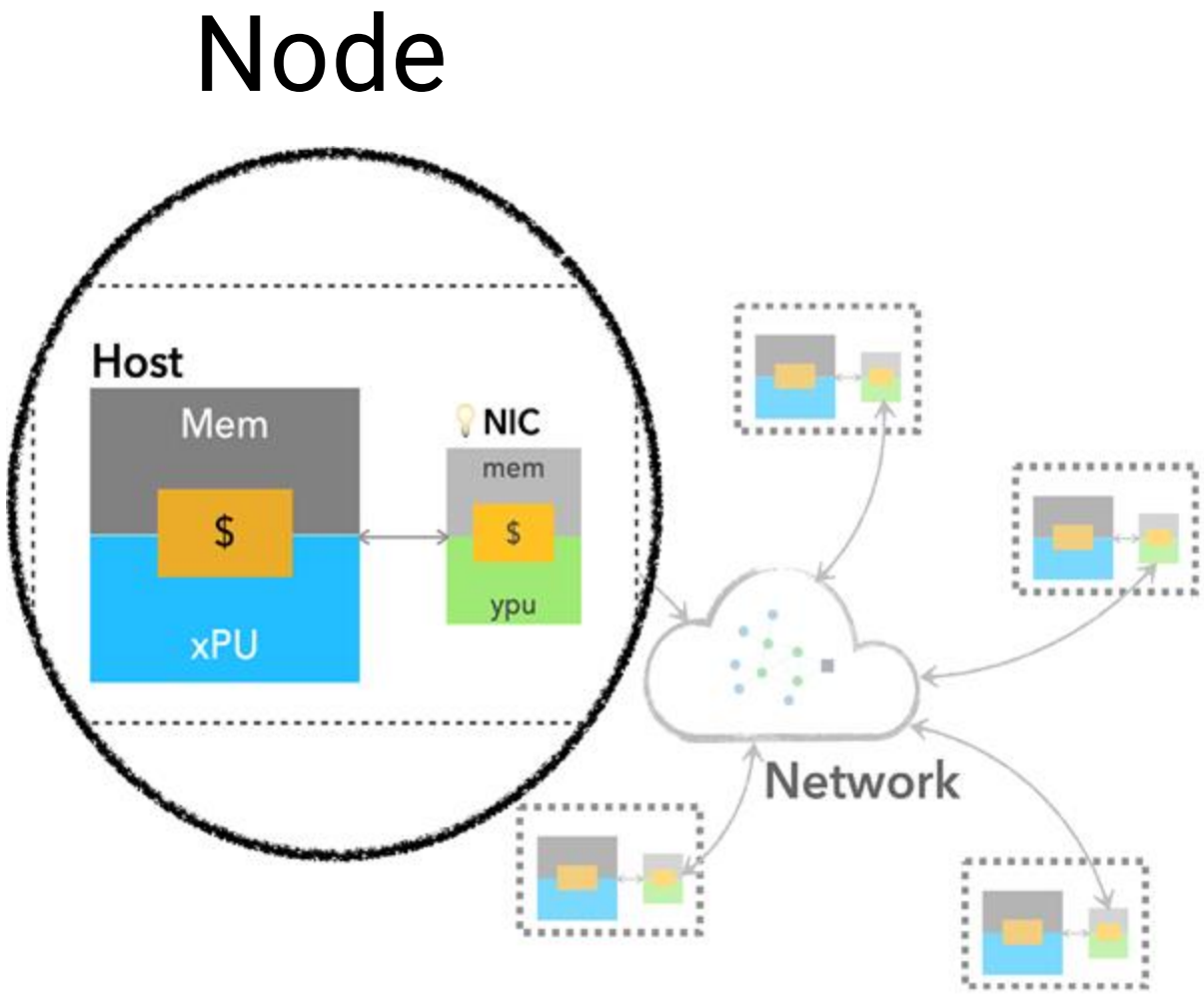
The basic building block of a distributed-memory cluster or supercomputer is a node.

Each node includes a host, which is a processor (xPU) + memory hierarchy.

The host can communicate with other hosts via its NIC (network interface controller).

A network connects the nodes. The nodes may be arranged in some topology, which determines the network's carrying capacity and cost.

In a **Data Processing Unit (DPU)**, the NIC becomes "host-like" via the addition of processing (ypu), memory and other accelerations engines.



Georgia Tech

How do we currently use SmartNICs?

Metadata management for storage systems

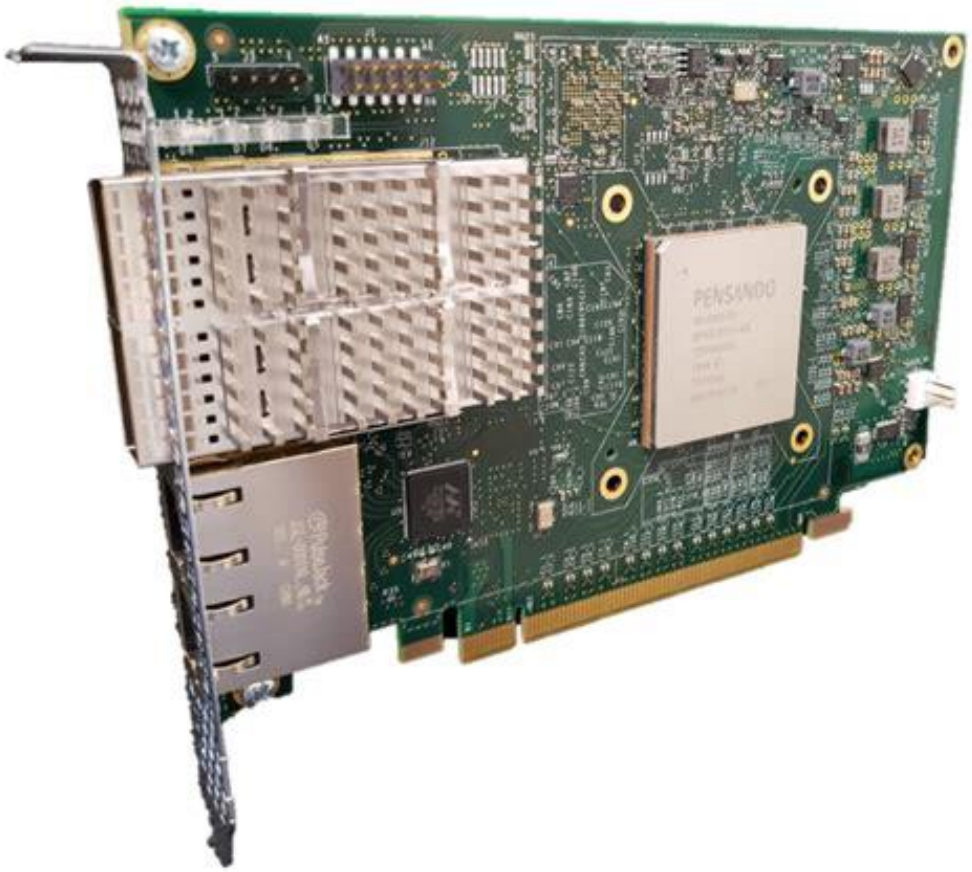
- Key part of VAST deployments

Containers for services

- DPU environment runs Linux and Kubernetes and can be deployed in a “Root of Trust” mode

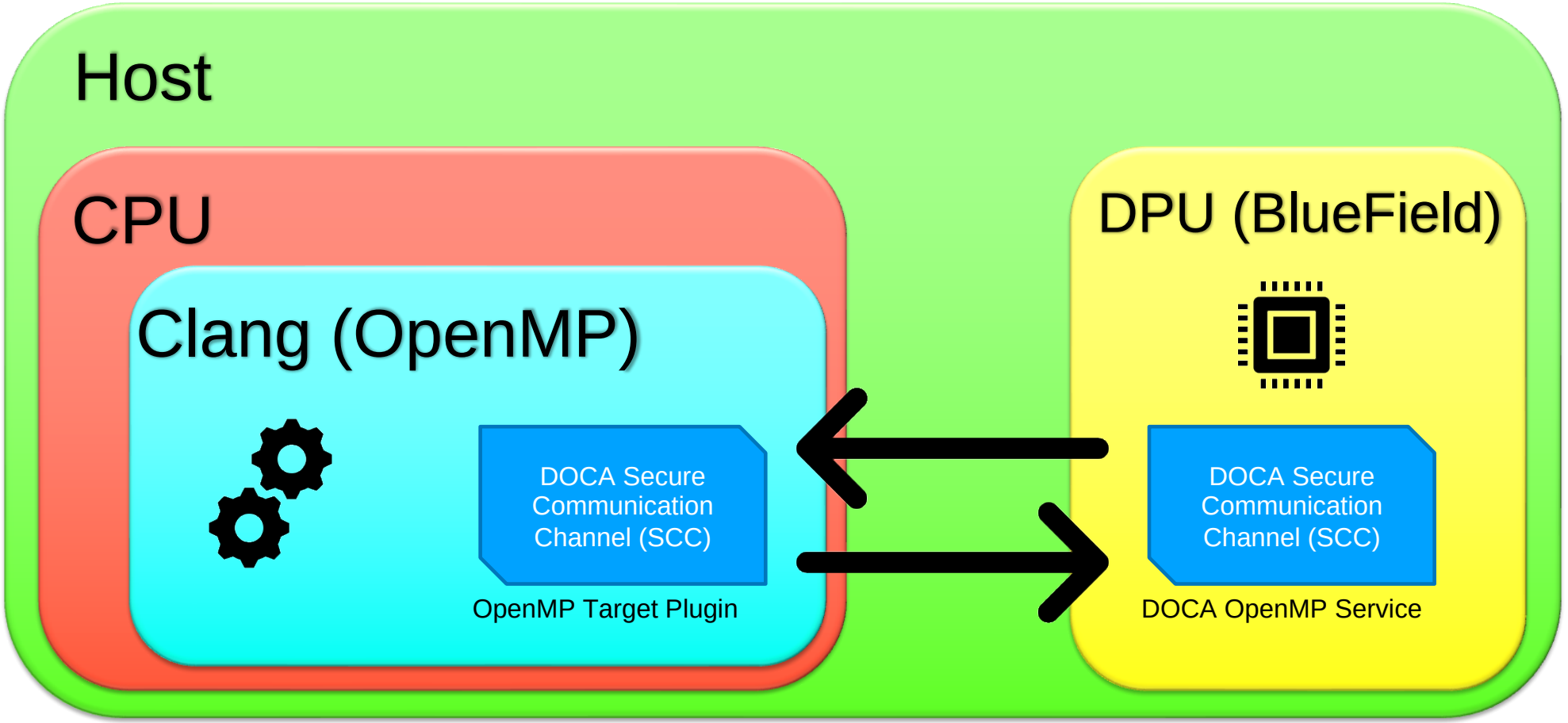
HPC experimentation and research

- OpenMP Offloading
- Algorithmic restructuring with BlueField SmartNICs
- Development of user-defined accelerators for FPGA-based SmartNICs



Georgia Tech

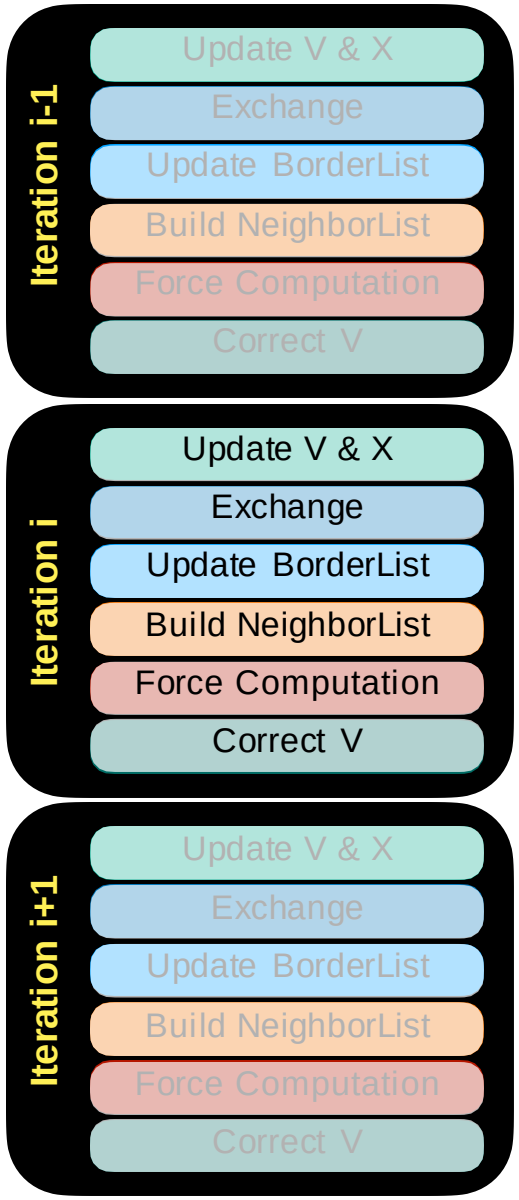
ODOS – Barcelona Supercomputing Center



See M. Usman, S. Iserte, R. Ferrer, and A. J. Peña, "DPU Offloading Programming with the OpenMP API." LLVM-HPC23 co-located with SC23, https://doi.org/10.1145/3624062.3624165

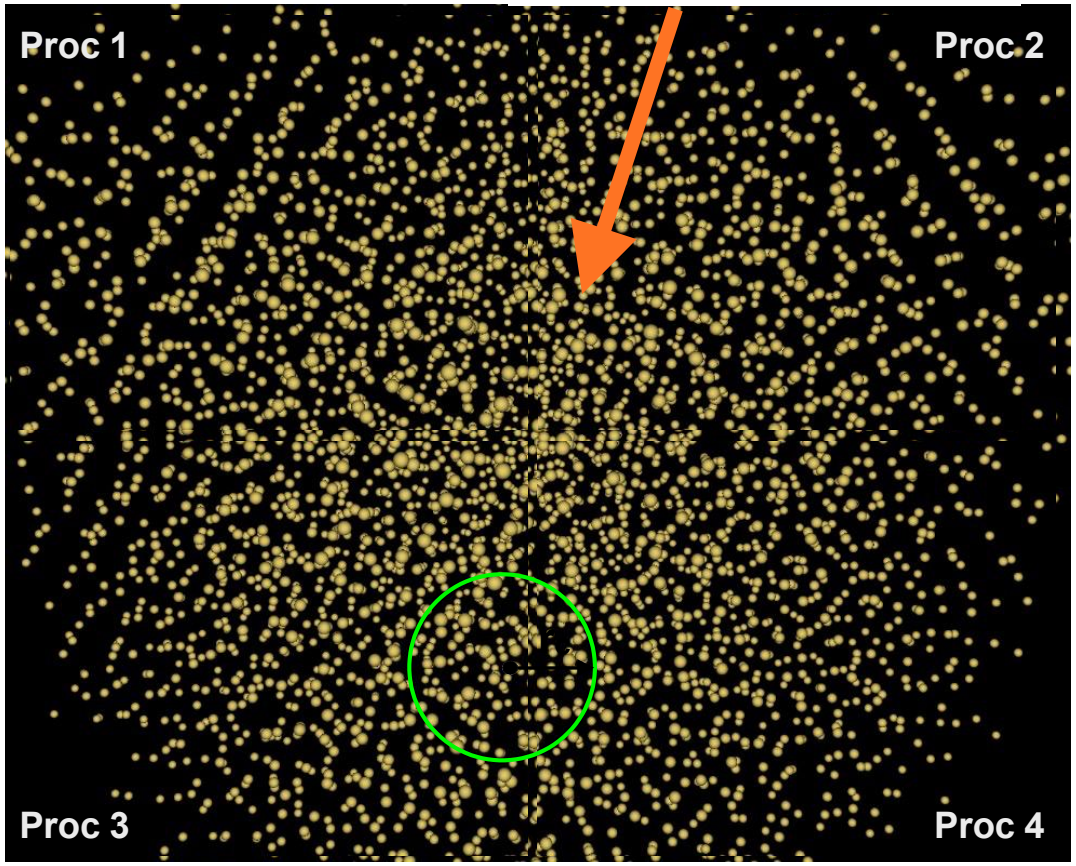
Georgia Tech

Using SmartNICs with MiniMD



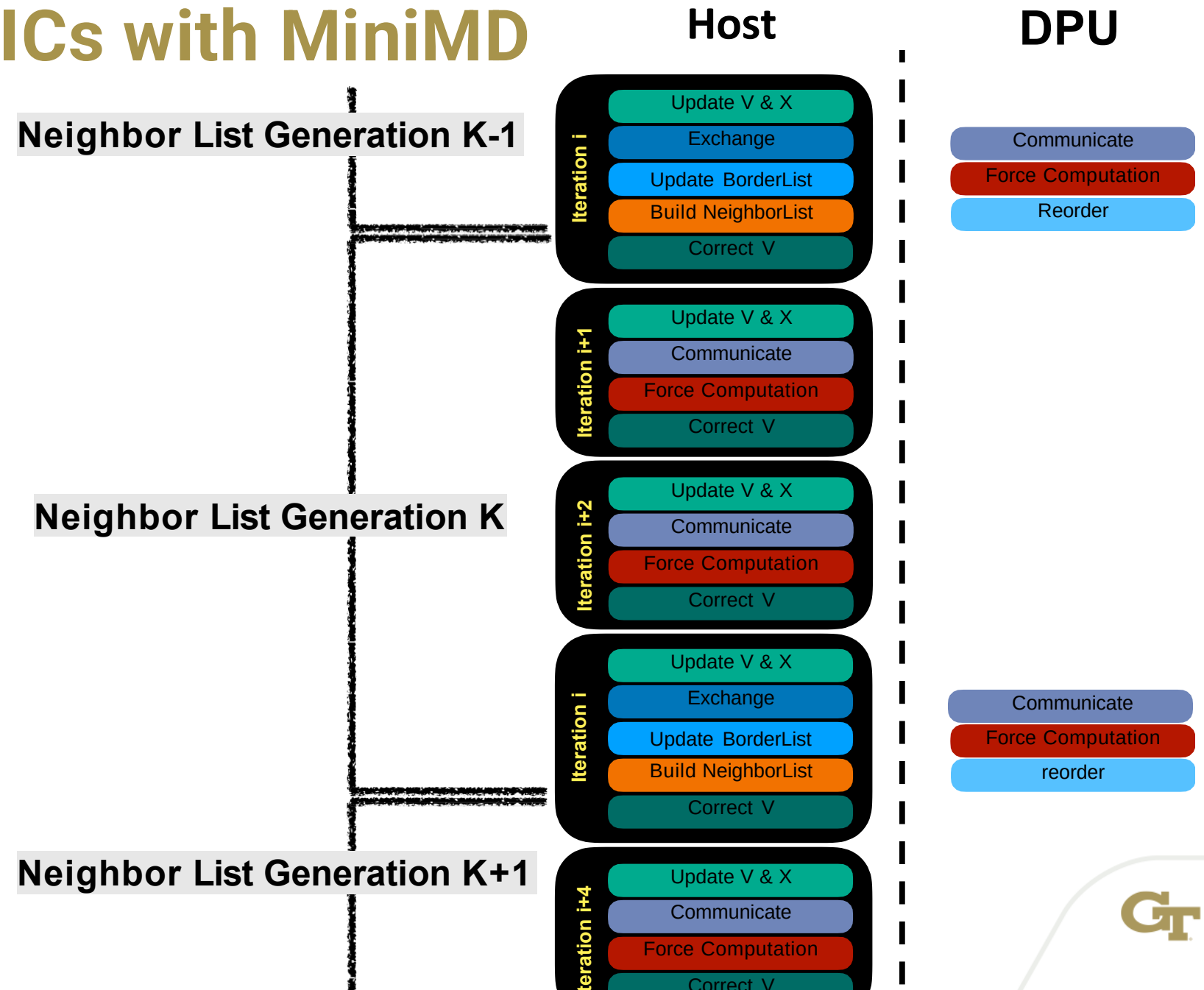
MiniMD is a molecular dynamics proxy-app. It calculates the position and velocity of a set of interacting particles in discrete time steps (iterations).

"Ghost zone" (comm+overhead)



S. Karamati et al., "Smarter" NICs for faster molecular dynamics: a case study, IPDPS, 2022

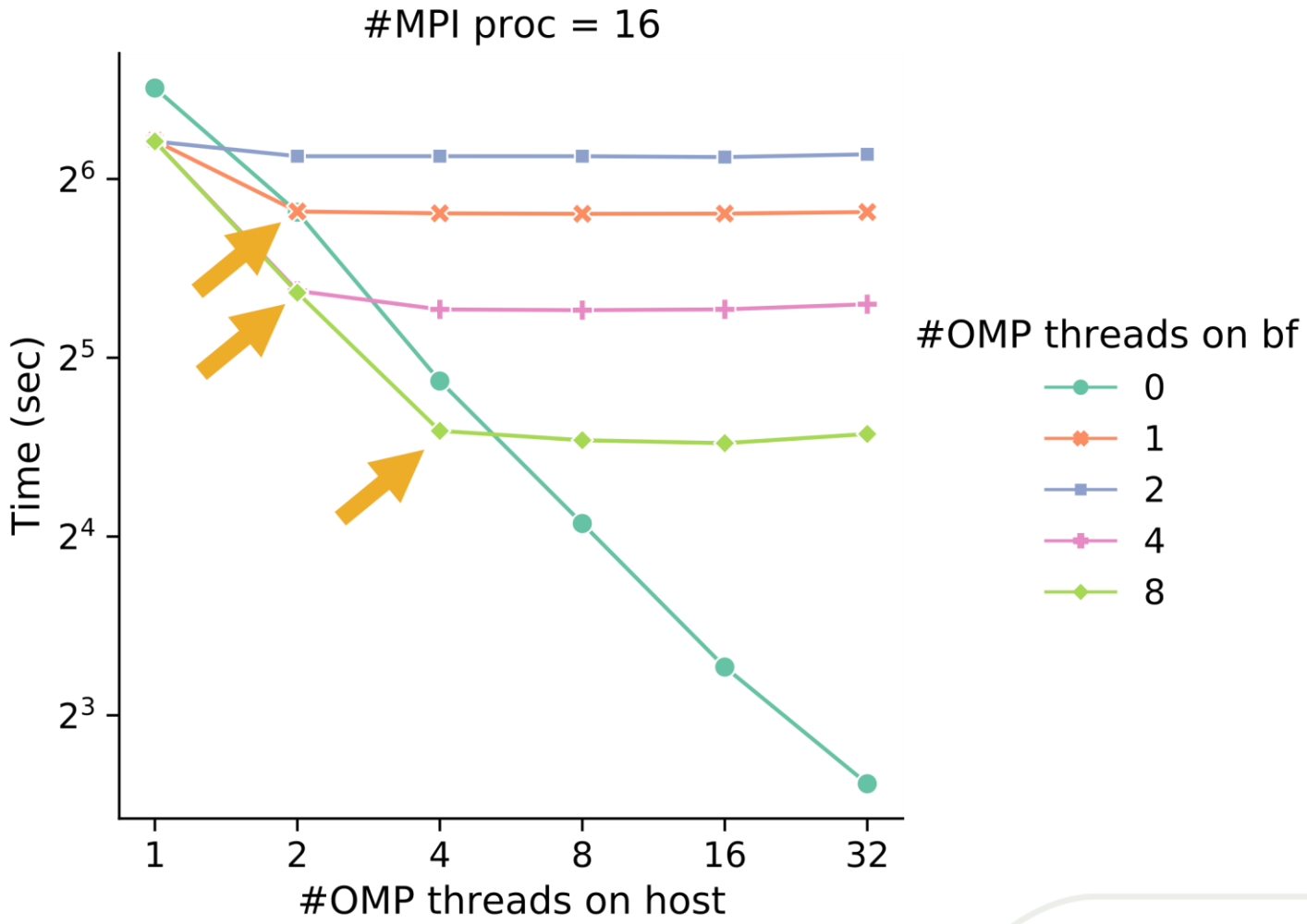
Using SmartNICs with MiniMD



Georgia Tech

MiniMD - Hybrid MPI/OpenMP performance results

- Our algorithm works best when it can completely hide the force computation time on BlueField.
- The degree of achievable overlap depends on the relative computational power of the host and BlueField.
- The knee of each curve indicates where the running times of neighbor-build on the host and force-compute on the BlueField are closest.



Georgia Tech

What should we be doing with DPUs?

HPC offload

- Collective communication offloading
- Offload for compute acceleration with ODOS
  - This assumes it is easy to test and show potential benefits for an application

HPC supporting services

- Containers that can help offload checkpointing or storage services for HPC applications

Secure HPC and AI computing

- Extend Trusted Execution Environments (TEEs) from one node to a larger storage/memory enclave

Prototyping Vehicle

- Test an idea for a HPC accelerator and then put it in hardware
- e.g., ZFP or compression/decompression accelerators
- Data Path Accelerators on BlueFields allow for “line-rate prototyping” without P4

What’s Next?

Georgia Tech

Follow-on Effects – Evaluating the Testbed

When we think about tomorrow's supercomputers, they are **highly heterogeneous**.

Testbeds like the Rogues Gallery allow us not only to evaluate far-off technologies but they are prototypes for near-term production systems

- Successful candidates migrate to our large-scale cluster, **Phoenix**, or the **Instructional Cluster Environment (ICE)**

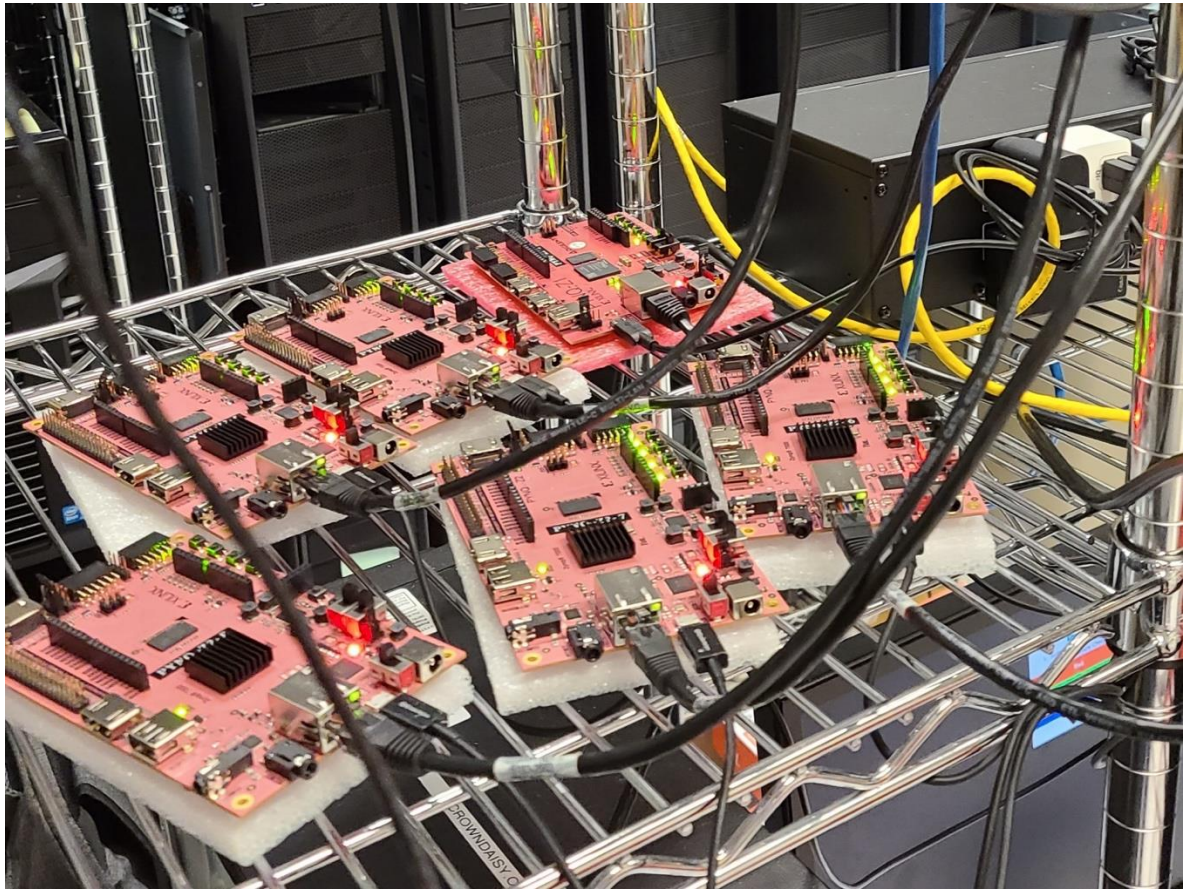
In addition to hardware, we prototype **software and tools** for deployment at scale

- Slurm features like node helpers and NSS support for launching cloud, container jobs
- Further adoption of user interfaces like Open OnDemand that make access easy so researchers can focus on the challenging systems and co-design tasks.

Georgia Tech

Bonus Example – Testbed + Production for Instruction

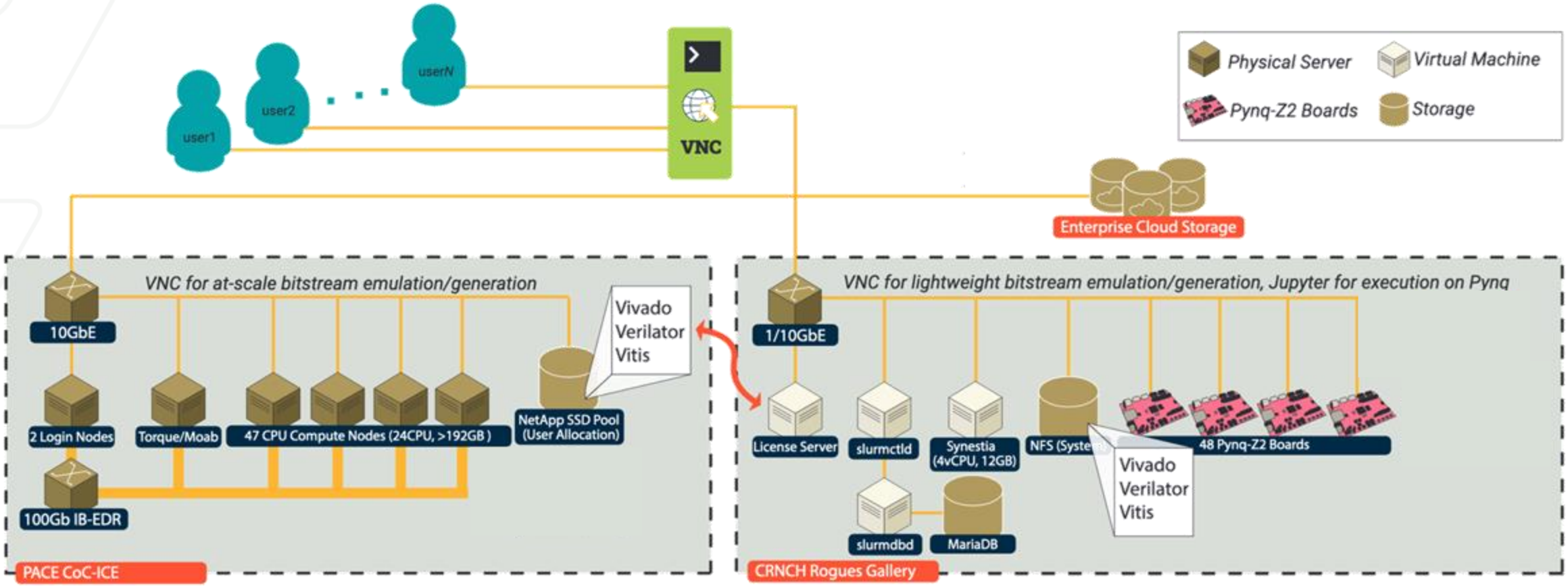
- Two courses with 90+ (CS 3220 and 40+ (ECE 8803) students
- Typical PYNQ FPGA workflows require:
  - The usage of Vivado and Vitis tools with large memory requirements
  - Hands-on interaction with a board to set up Jupyter webserver interfaces
- Supply chain issues limited the number of boards we could deploy



How can we best support remote FPGA development and usage at scale?

Georgia Tech

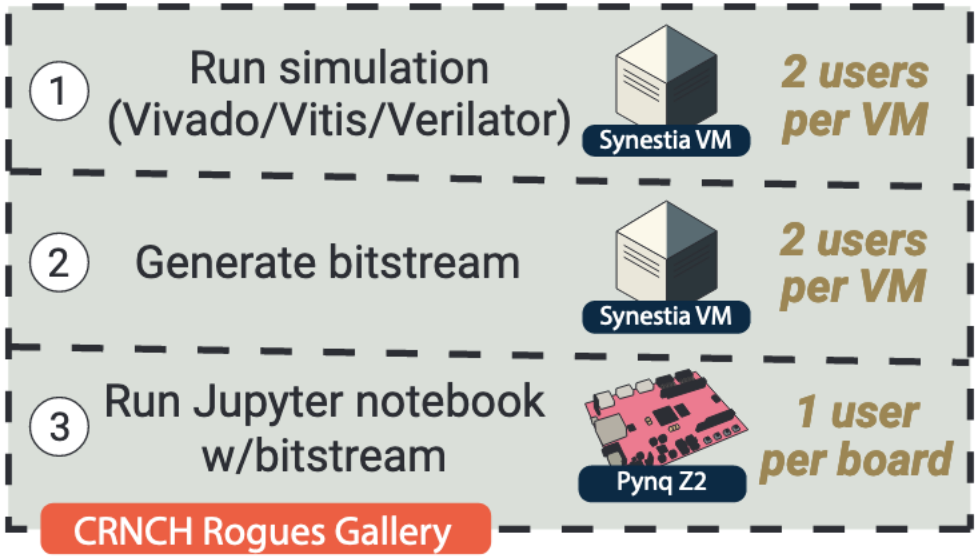
CRNCH + PACE Instructional Workflow

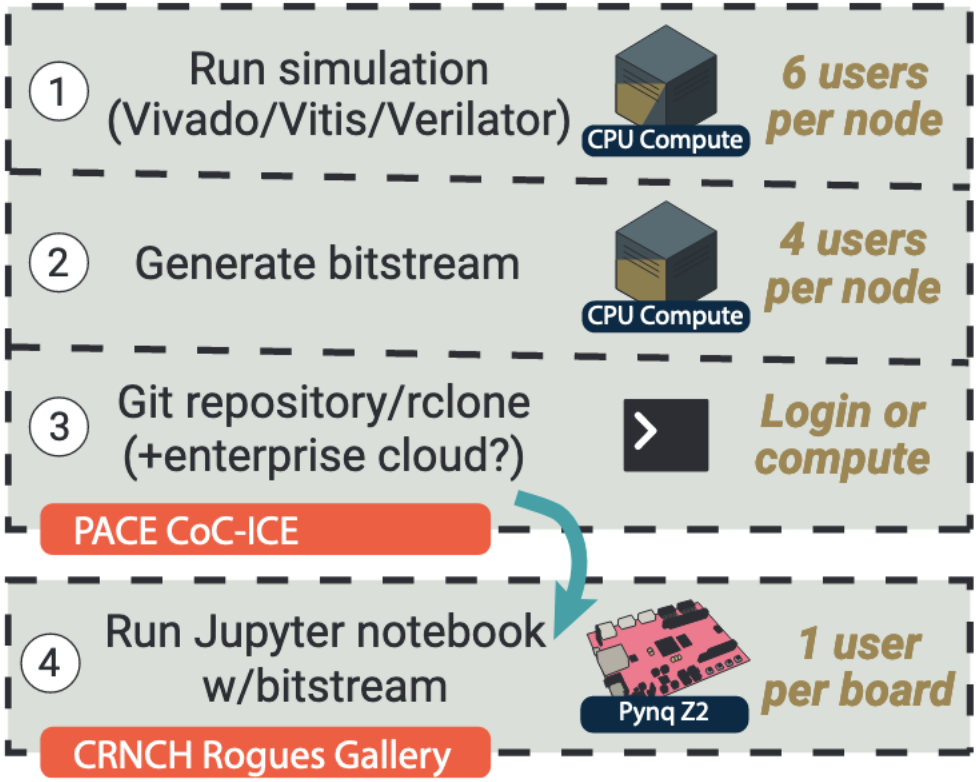


See our SEHET22 paper, “Enhancing HPC Education and Workflows with Novel Computing Architectures” at https://github.com/gt-crnch-rg/rg-publications/

Georgia Tech

Two User Workflows





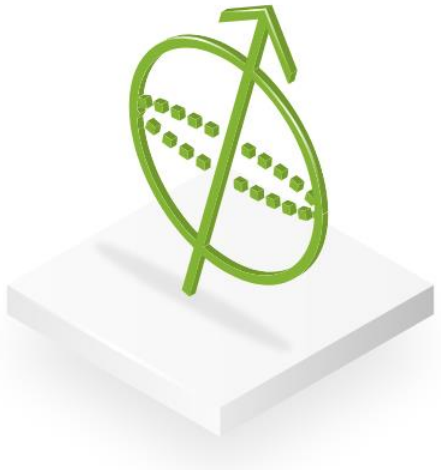
A combined workflow allows for a dramatic reduction in the bottleneck of design and initial testing

- However, there are some issues to consider related to data security
- ICE is set up to manage student data unlike research testbeds like CRNCH RG

Georgia Tech

Integration with Institutional Computing

Application portability across clusters through containerized workflows!



Chisel

PACE resources (Phoenix and ICE) provide more capacity and capability, so as problems outgrow RG, extend workflows accordingly

- Xilinx libraries on ICE for FPGA development
- Chisel/Chipyard containers on Phoenix
- CuQuantum containers and support

Georgia Tech

Heterogeneous AI Workflows

1 Common Storage

Common storage (CEDAR, IDEaS) across campus allow opportunity to migrate data/models to optimal architectures.

Toolkits for exploring alternative architectures and pipelines to evaluate appropriate hardware before production

2 Simulate and Dispatch

3 Function as a Service

Globus Compute Endpoint (funcX)/CyberShuttle as a means to target different accelerators seamlessly within a single application.

Georgia Tech

Heterogeneous User Groups

Temporary user access groups

- Single lab assignment, workshops, or tutorials
- Makes use of JupyterHub, Slurm job container plugins, topic-specific containers
- Requires additional security technical and policy discussions

Medium-term users of resources for instructional or research purposes

- Semester-long courses or research experiences
- May require longer-term access; can still use Slurm job container plugins, topic-specific containers

General users

- Long-term research or instructional accounts; service accounts
- Full user-level access
- Often require specialized support or limited sudo to run some commands (‘nvidia-smi’ or “docker run’)

Georgia Tech

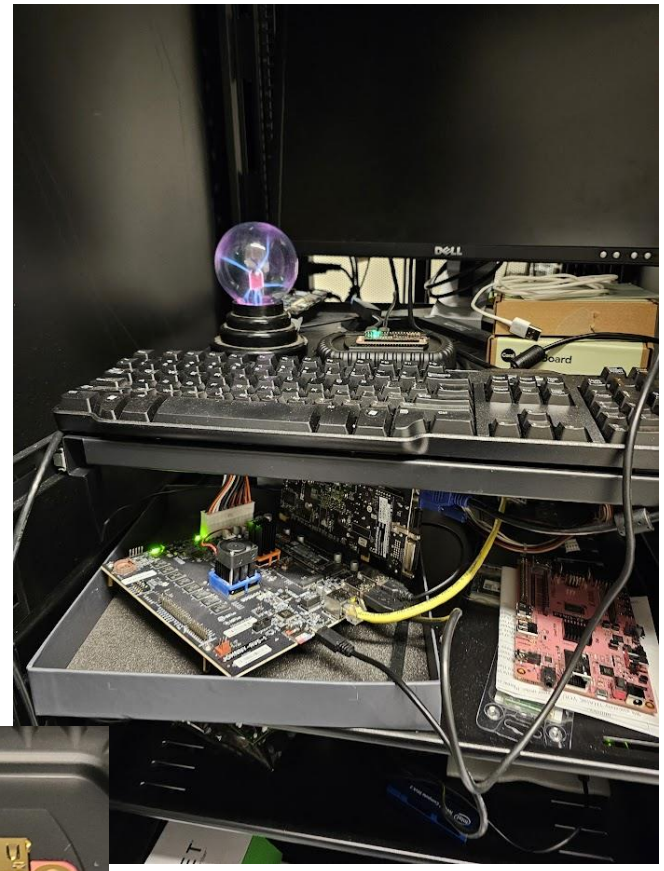
Challenges in CRNCH RG Deployment

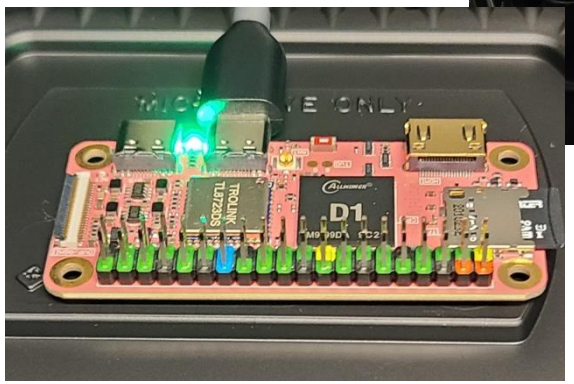
Supporting one class of novel architecture requires an enormous investment in backend resources, training, and infrastructure

It's hard to imagine exerting the same level of effort for every new platform!

e.g., Currently, RG uses 16 different builds of Slurm, which is only manageable with automation

However, recognizing the patterns for typical novel architecture workflows and mapping them to common infrastructure can provide a solid basis for research





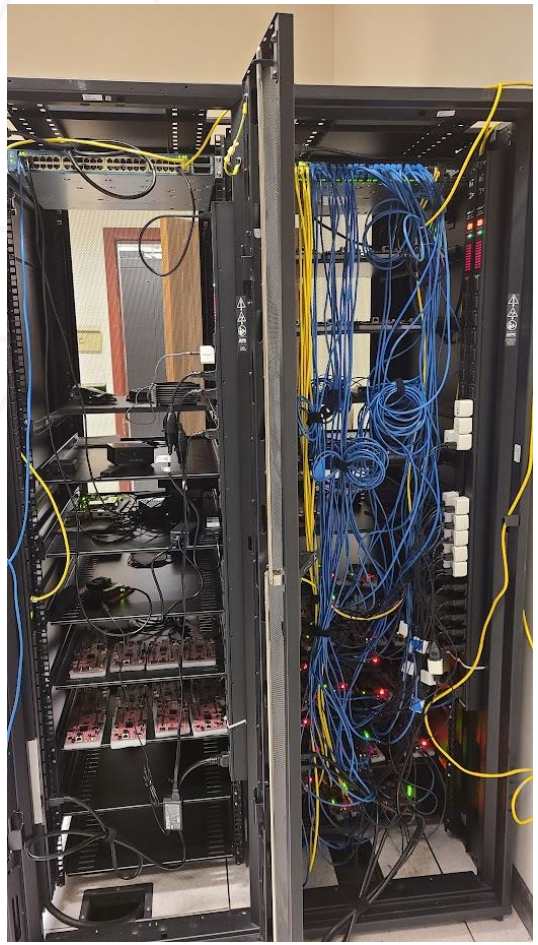
Georgia Tech

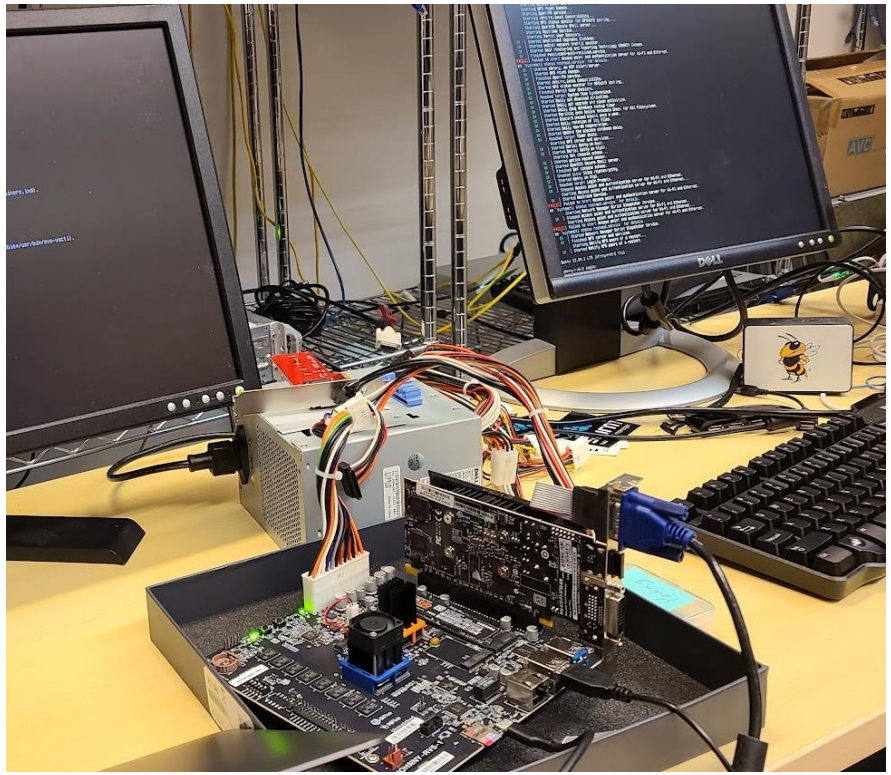
Summary

Novel architectures will be critical to understanding data movement trends and designing solutions for post-exascale systems

- Device heterogeneity is here to stay and it may be critical for future performance!
- Testbeds can be a key vehicle for both this research and related instruction and training
- Getting the proper support for a platform is critical to its success
  - Each “candidate architecture” needs enthusiast users and dedicated researchers invested in research success
  - Courses and applications that can make use of a unique feature are also good targets for development and usage of novel architectures
  - Training and high-level tools are vital to reduce barriers to usage

Questions?





Project site: crnch-rg.cc.gatech.edu
GH Presence: https://github.com/gt-crnch-rg

Apply for access: https://crnch-rg.cc.gatech.edu/request-rogues-gallery-access/

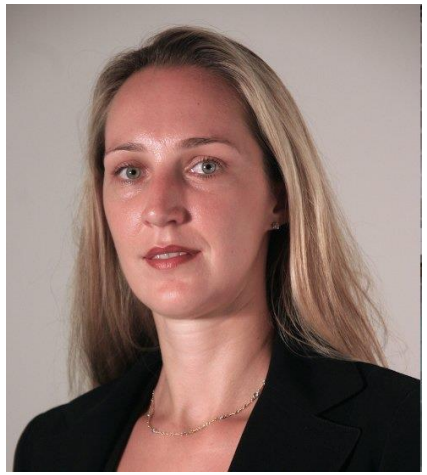
Georgia Tech

Many Thanks to the CRNCH and RG Contributors

NSF Project Co-PIs

Tom Conte

Jennifer Hasler



Ada Gavrilovska

Rich Vuduc

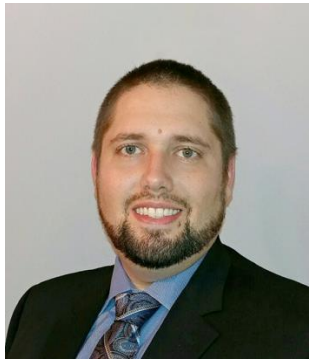
Senior Personnel

Alex Daglis

Technical Support

Sterling Peet

Will Powell



Aaron Jezghani (PACE)

Georgia Tech

Acknowledgements

This work is supported by the NSF Rogues Gallery testbed grant (CNS-2016701)

This research used resources provided by the Darwin testbed at Los Alamos National Laboratory (LANL) which is funded by the Computational Systems and Software Environments subprogram of LANL's Advanced Simulation and Computing program (NNSA/DOE).

Kevin Sheridan, Galen Shipman, and Jered Dominguez-Trujillo acknowledge support by the National Nuclear Security Administration. Los Alamos National Laboratory is operated by Triad National Security, LLC for the U.S. Department of Energy under contract 89233218CNA000001; LA-UR-24-24856.

Georgia Tech