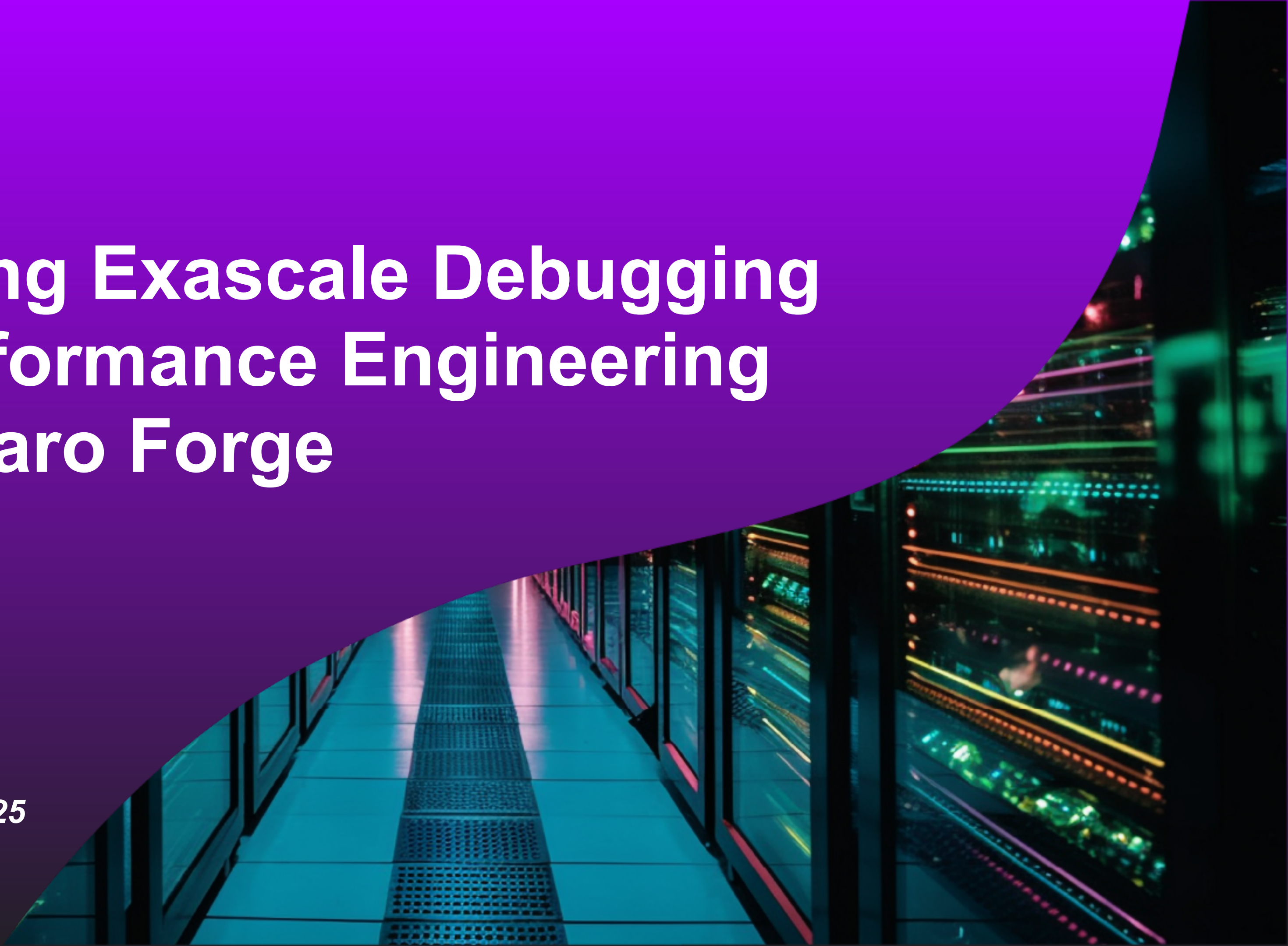
linaroforge

Unlocking Exascale Debugging and Performance Engineering with Linaro Forge

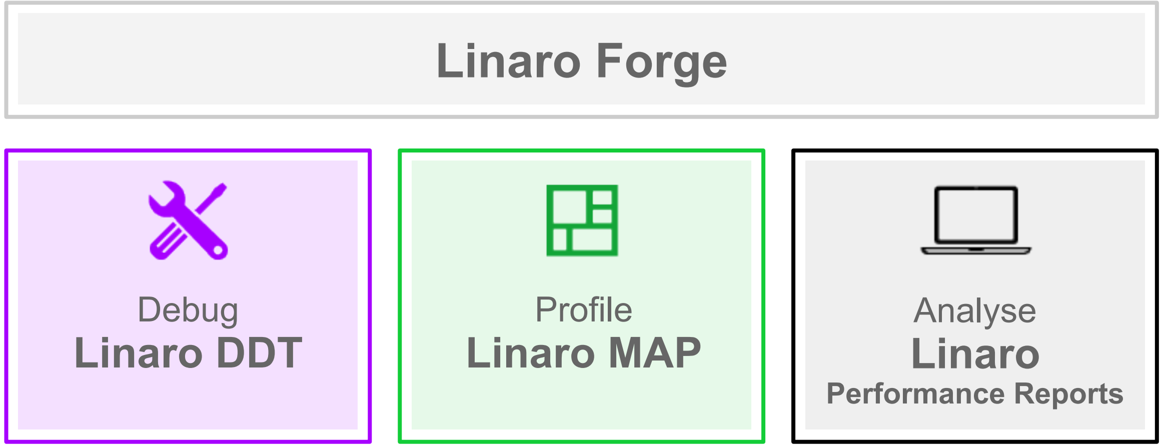
Rudy Shand
Principal FAE
Linaro

CUG 2025
May 04 - May 08, 2025
New York



HPC Development Solutions from Linaro

Best in class commercially supported tools for Linux and high-performance computing (HPC)



Linaro Forge

Debug
Linaro DDT

Profile
Linaro MAP

Analyse
Linaro
Performance Reports

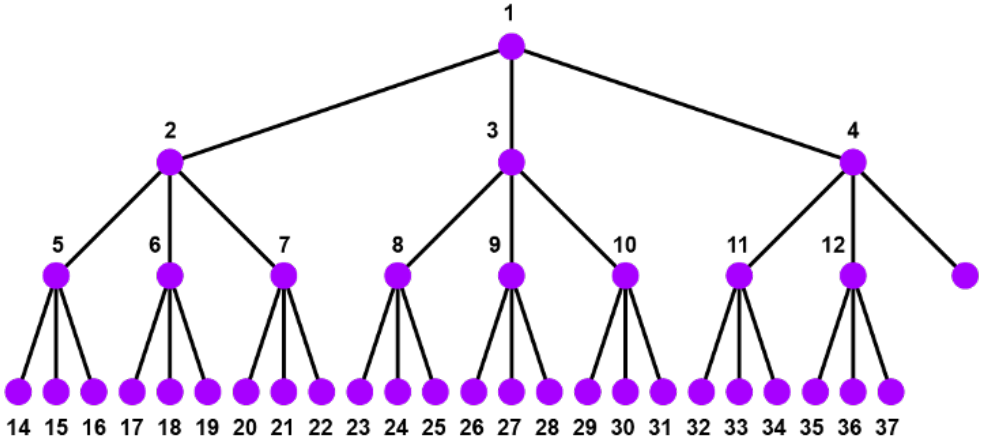
Performance Engineering for any architecture, at any scale

Scalable Tree Network

Unlocking Exascale Debugging and Performance Engineering

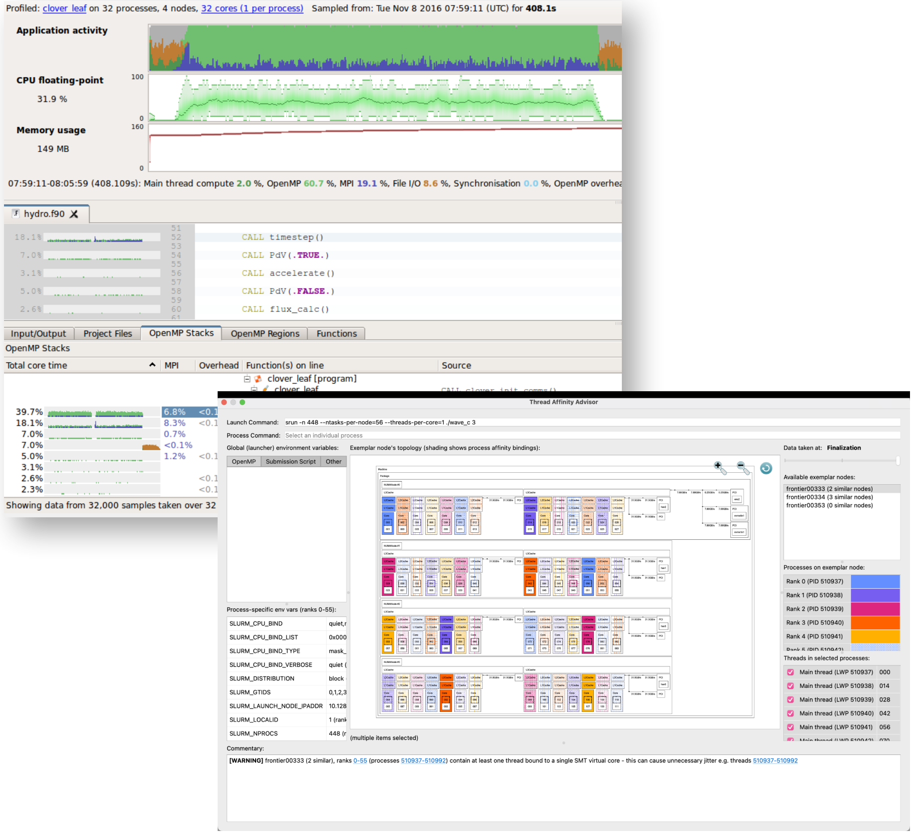
- Treeserver design is how is how Forge is able to scale across processes
- Send bulk commands and merge responses
- Aggregate the data instead of broadcasting hundreds / thousands responses

Tree network topology



linaroforge

Debugging and Profiling to exascale and beyond!



Profiled: clover_leaf on 32 processes, 4 nodes, 32 cores (1 per process) Sampled from: Tue Nov 8 2016 07:59:11 (UTC) for **408.1s**

Application activity	
CPU floating-point	31.9 %
Memory usage	149 MB

07:59:11-08:05:59 (408.109s): Main thread compute 2.0 %, OpenMP 60.7 %, MPI 19.1 %, File I/O 8.6 %, Synchronisation 0.0 %, OpenMP overhea

hydro.f90

```
18.1%       CALL timestep()
 7.0%       CALL PdV(.TRUE.)
 3.1%       CALL accelerate()
 5.0%       CALL PdV(.FALSE.)
 2.6%       CALL flux_calc()
```

OpenMP Stacks

Total core time	MPI	Overhead	Function(s) on line	Source
			clover_leaf [program]	
			clover_leaf	CALL clover_init_comms()
39.7%	6.8%	<0.1		
18.1%	8.3%	<0.1		
7.0%	0.7%			
7.0%	<0.1%			
5.0%	1.2%	<0.1		
3.1%				
2.6%		<0.1		
2.3%		<0.1		

Showing data from 32,000 samples taken over 32

Thread Affinity Advisor

Launch Command: srun -n 448 --ntasks-per-node=56 --threads-per-core=1 ./wave_c 3

Process Command: Select an individual process

Global (launcher) environment variables:

Exemplar node's topology (shading shows process affinity bindings):

(multiple items selected)

Data taken at: **Finalization**

Available exemplar nodes:

frontier00333 (2 similar nodes)
frontier00334 (3 similar nodes)
frontier00353 (0 similar nodes)

Process-specific env vars (ranks 0-55):

SLURM_CPU_BIND	quiet,r
SLURM_CPU_BIND_LIST	0x000
SLURM_CPU_BIND_TYPE	mask_
SLURM_CPU_BIND_VERBOSE	quiet (
SLURM_DISTRIBUTION	block
SLURM_GTIDS	0,1,2,3
SLURM_LAUNCH_NODE_IPADDR	10.128
SLURM_LOCALID	1 (rank
SLURM_NPROCS	448 (r

Processes on exemplar node:

Rank 0 (PID 510937)
Rank 1 (PID 510938)
Rank 2 (PID 510939)
Rank 3 (PID 510940)
Rank 4 (PID 510941)
Rank 5 (PID 510942)

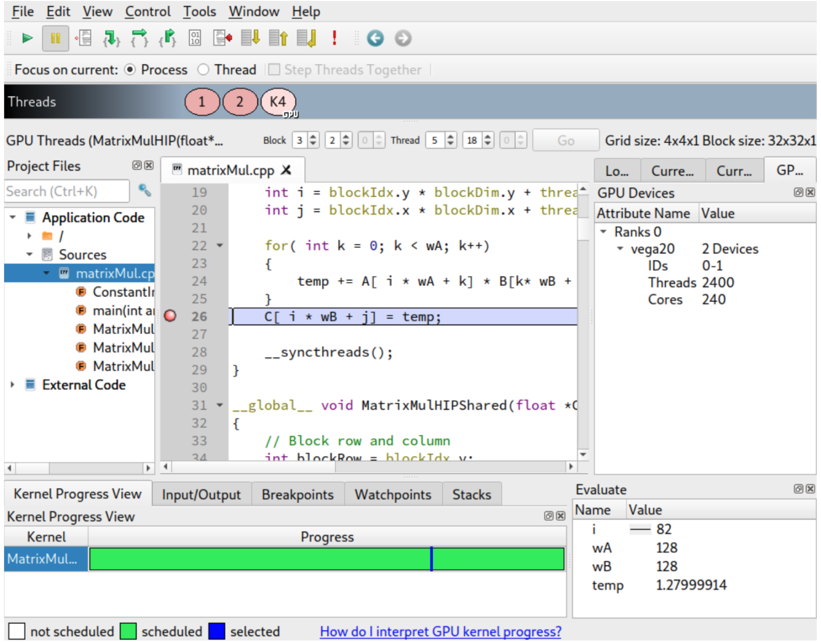
Threads in selected processes:

☑	Main thread (LWP 510937)	000
☑	Main thread (LWP 510938)	014
☑	Main thread (LWP 510939)	028
☑	Main thread (LWP 510940)	042
☑	Main thread (LWP 510941)	056
☑	Main thread (LWP 510942)	070

Commentary:

[WARNING] frontier00333 (2 similar), ranks 0-55 (processes 510937-510992) contain at least one thread bound to a single SMT virtual core - this can cause unnecessary jitter e.g. threads 510937-510992

- Highly scalable architecture
- Rich set of features
- Cross platform



Focus on current: ● Process ○ Thread □ Step Threads Together

GPU Threads (MatrixMulHIP(float*... Block 3 2 0 Thread 5 18 0 Go Grid size: 4x4x1 Block size: 32x32x1

Project Files

Search (Ctrl+K)

- Application Code
  - /
  - Sources
    - matrixMul.cp
      - ConstantI
      - main(int a
      - MatrixMul
      - MatrixMul
      - MatrixMul
- External Code

matrixMul.cpp

```
    int i = blockIdx.y * blockDim.y + threa
    int j = blockIdx.x * blockDim.x + threa

    for( int k = 0; k < wA; k++)
    {
        temp += A[ i * wA + k] * B[k* wB +
    }
    C[ i * wB + j] = temp;

    __syncthreads();
}

__global__ void MatrixMulHIPShared(float *C
{
    // Block row and column
    int blockRow = blockIdx.y;
```

GPU Devices

Attribute Name	Value
Ranks 0	
vega20	2 Devices
IDs	0-1
Threads	2400
Cores	240

Kernel Progress View

Kernel	Progress
MatrixMul...	

□ not scheduled 🟩 scheduled 🟦 selected How do I interpret GPU kernel progress?

Evaluate

Name	Value
i	82
wA	128
wB	128
temp	1.27999914

Forge Highlights

Built from the ground up for unlimited scalability

Long relationship with the Cray/HPE teams to ensure flawless operation on leadership class machines

Support for C/C++, FORTRAN, and Python based codes

Debugging and Profiling Traditional HPC & Machine Learning codes

Cross platform support for x86, and arm cpu architectures, and Nvidia, AMD, and Intel gpu systems

Thread Affinity Advisor, Cray CPU Energy Metrics, MAP / PR Diff tool

Thank you

Go to www.linaroforge.com