Linaro Forge

Debugging and Optimization Tools for HPC

Linaro Forge

Agenda

- A Brief history
- DDT Overview (Debugger)
- MAP Overview (Profiler)
- Performance Reports Overview

A Brief History

2014: Release of Allinea tools 5.0, with addition of the new Allinea Performance Reports.

December 2016: Arm extends HPC offering with acquisition of software tools provider Allinea Software.

12 major releases

30th January 2023: Linaro to Acquire Arm Forge Software Tools Business.

HPC Development Solutions from Linaro

Best in class commercially supported tools for HPC

Linaro Forge

Debug	Profile	Analyse
Linaro DDT	**Linaro MAP**	**Linaro Performance Reports**

Performance Engineering for any architecture, at any scale

Supported Platforms

Intel Compiler, ROCm, CCE, ACfL, GCC, NVHPC, IBM XL	Compiler; Python
Intel MPI, HPE MPI, MPICH, Open MPI, …, IBM Spectrum MPI	Slurm; PALS
RHEL 7+, SLES 15, Ubuntu 20.04+	macOS; Windows
AMD ROCm, NVIDIA CUDA	GP-GPU Accelerator
AMD/Intel (x86-64), Arm (AArch64), Power8 (ppc64le)	CPU Architecture

Linaro Forge

An interoperable toolkit for debugging

The de-facto standard for HPC development

- Most widely-used debugging and profiling suite in HPC
- Fully supported by Linaro on Intel, AMD, Arm, Nvidia, AMD GPUs, etc.

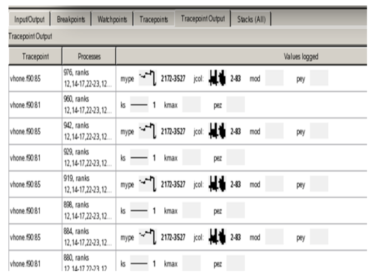
State-of-the art debugging capabilities

- Powerful and in-depth error detection mechanisms (including memory debugging)
- Available at any scale (from serial to exascale applications)

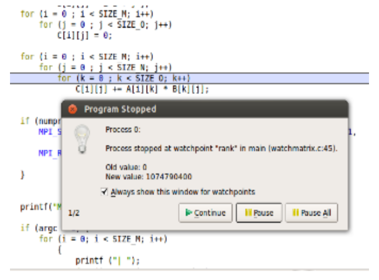
Easy to use by everyone

- Unique capabilities to simplify remote interactive sessions
- Innovative approach to present quintessential information to users

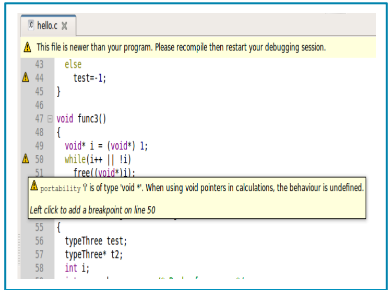
Linaro DDT Debugger Highlights



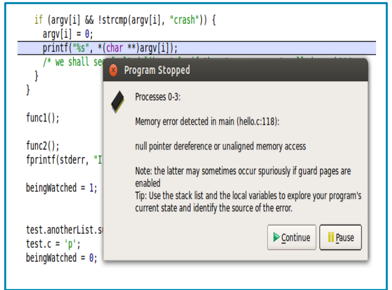
The scalable print alternative



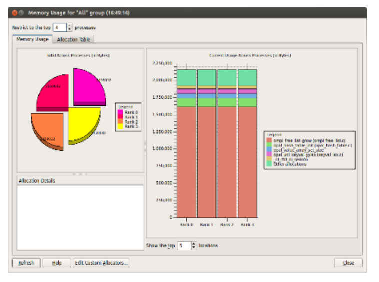
Stop on variable change



Static analysis warnings on code errors



Detect read/write beyond array bounds



Detect stale memory allocations

Linaro Forge

Multi-dimensional Array Viewer

What does your data look like at runtime?

View arrays

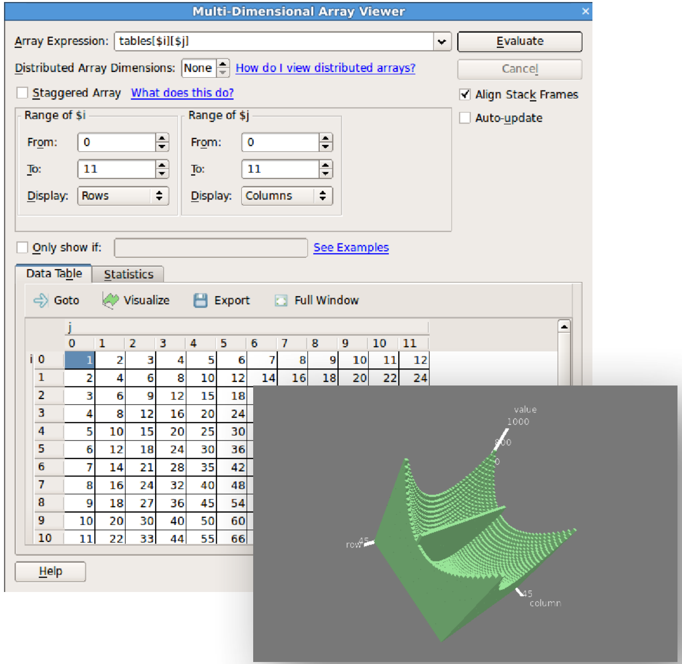
- On a single process
- Or distributed on many ranks

Use metavariables to browse the array

- Example: $i and $j
- Metavariables are unrelated to the variables in your program
- The bounds to view can be specified
- Visualise draws a 3D representation of the array

Data can also be filtered

- “Only show if”: $value>0 for example $value being a specific element of the array



The Performance Roadmap

Optimizing high performance applications

Improving the efficiency of your parallel software holds the key to solving more complex research problems faster.

This pragmatic, 9 Step best practice guide, will help you identify and focus on application readiness, bottlenecks and optimizations one step at a time.

Bugs

- Correct application

Analyze before you optimize

- Measure all performance aspects. You can't fix what you can't see.
- Prefer real workloads over artificial tests.

I/O

- Discover lines of code spending a long time in I/O.
- Trace and debug slow access patterns.

Workloads

- Detect issues with balance.
- Slow communication calls and processes.

Dive into partitioning code.

Communication

- Track communication performance.
- Discover which communication calls are slow and why.

Memory

- Reveal lines of code bottlenecked by memory access times.
- Trace allocation and use of hot data structure

Cores

- Discover synchronization overhead and core utilization
- Synchronization-heavy code and implicit barriers are revealed

Vectorization

- Understand numerical intensity and vectorization level.
- Hot loops, unvectorized code and GPU performance reveleaed

Verification

- Validate corrections and optimal performance

Key :
- (blue) Linaro Forge
- (orange) Linaro Performance Reports

Linaro Performance tools

Characterize and understand the performance of HPC application runs

Commercially supported by Linaro

Gather a rich set of data

- Analyses metric around CPU, memory, IO, hardware counters, etc.
- Possibility for users to add their own metrics

Accurate and Astute insight

Build a culture of application performance & efficiency awareness

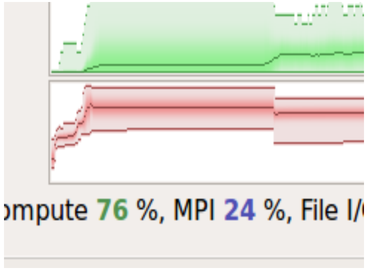
- Analyses data and reports the information that matters to users
- Provides simple guidance to help improve workloads' efficiency

Relevant advice to avoid pitfalls

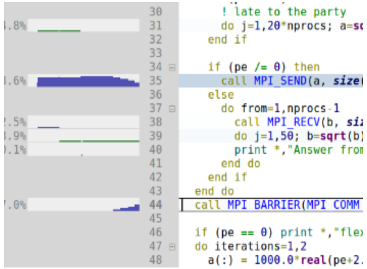
Adds value to typical users' workflows

- Define application behaviour and performance expectations
- Integrate outputs to various systems for validation (eg. continuous integration)
- Can be automated completely (no user intervention)

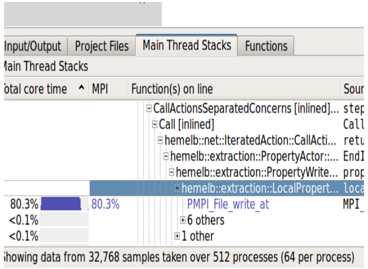
Linaro MAP Source Code Profiler Highlights



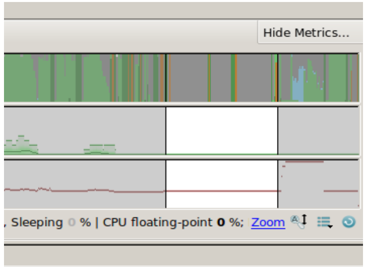
Find the peak memory use



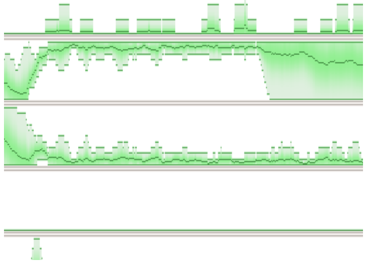
Fix an MPI imbalance



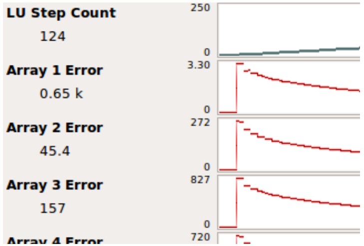
Remove I?O bottleneck



Make sure OpenMP regions make sense



Improve memory access



Custom Metrics

MAP Capabilities

MAP is a sampling based scalable profiler

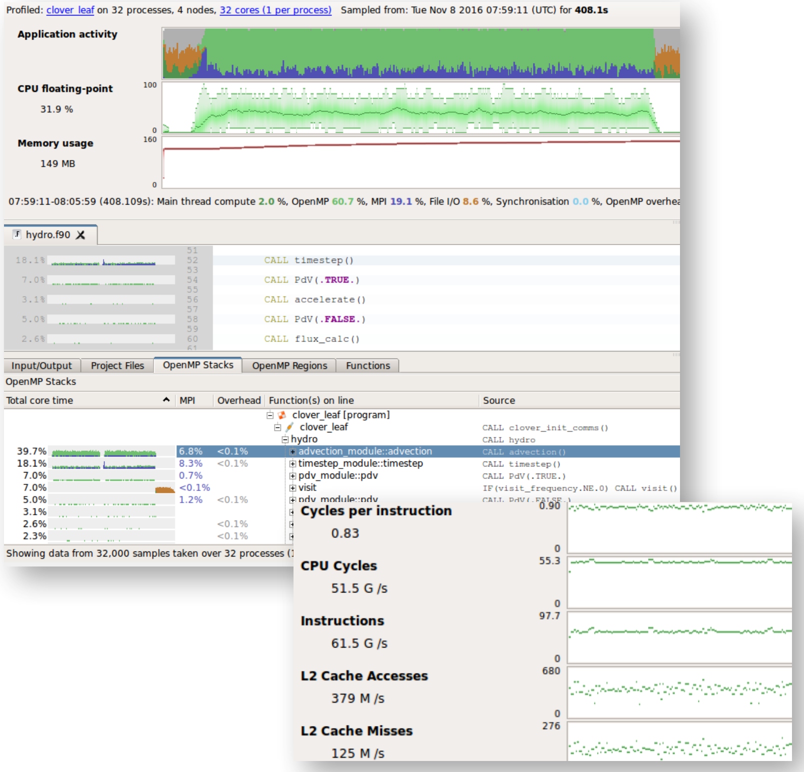
- Built on same framework as DDT
- Parallel support for MPI, OpenMP, CUDA
- Designed for C/C++/Fortran

Designed for 'hot-spot' analysis

- Stack traces
- Augmented with performance metrics

Adaptive sampling rate

- Throws data away - 1,000 samples per process
- Low overhead, scalable and small file size



Thank you

Go to www.linaroforge.com

rudy.shand@linaro.org

Linaro Forge

Hands on examples

Install Forge https://www.linaroforge.com/downloadForge

Forge user guide https://docs.linaroforge.com/23.0.1/html/forge/forge/index.html

/grand/ATPESC2023/Linaro-Forge/examples

Installed as part of Forge tools as well
<forge location>/examples

Use the temporary license shown below

export ALLINEA_FORCE_LICENCE_FILE=/grand/ATPESC2023/Linaro-Forge/Licence.trial

Remote client cheat sheet

Install the Remote Client

https://www.linaroforge.com/downloadForge

Setup the client

1. Open your Remote Client
2. Create a new connection:RemoteLaunch➔Configure➔Add
3. Hostname: *<username>@theta.alcf.anl.gov*
4. Remote installation directory: /soft/debuggers/forge-22.0.4-2022-08-02

Setup the remote side

1. qsub -I -n 8 -A ATPESC2023 -q debug-cache-quad -t 30 --attrs filesystems=home,grand,eagle
2. module load forge
3. module unload xalt
4. module unload darshan/3.3.0
5. ddt --connect --mpi="Cray XT/XE/XK (MPI/shmem)" aprun -n 8 ./hello_c

Debugging on Thetagpu

The latest Forge modules are not available on thetagpu, but you can you use the installed software directly

Debug your GPU code using:
ddt --connect gpu_code.exe

Profiling on Theta

Although static binaries are created by default on Theta, it is recommended to build dynamic executables for profiling purposes with the compiler flag **-dynamic**

If you get library missing errors, reload the intel module

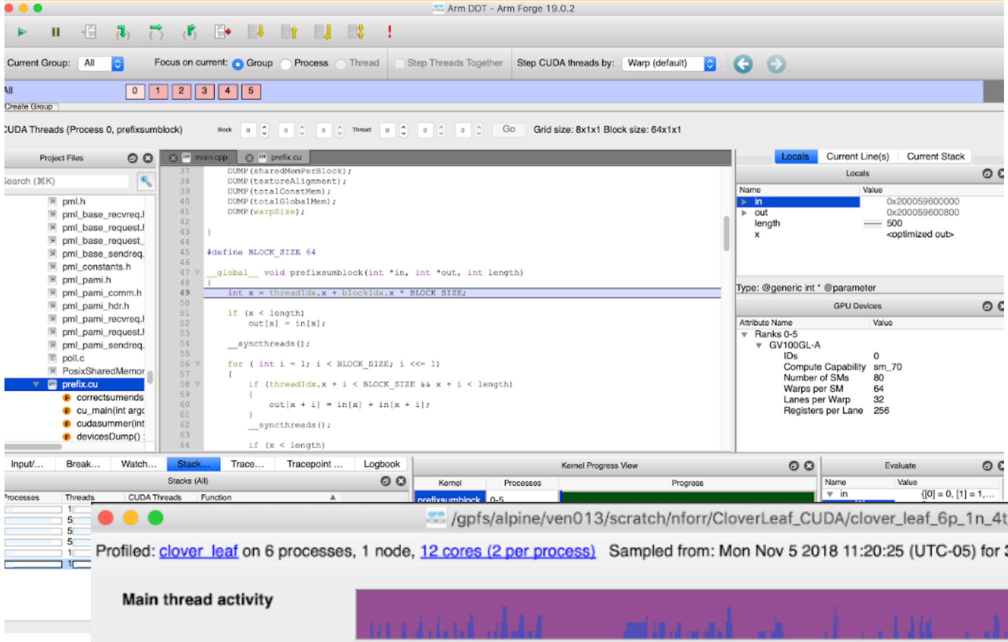
moduleunloadintel

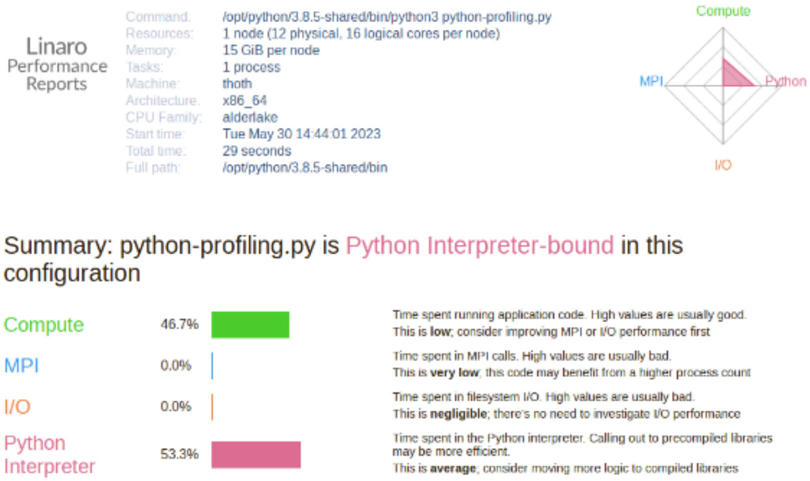
moduleloadintel

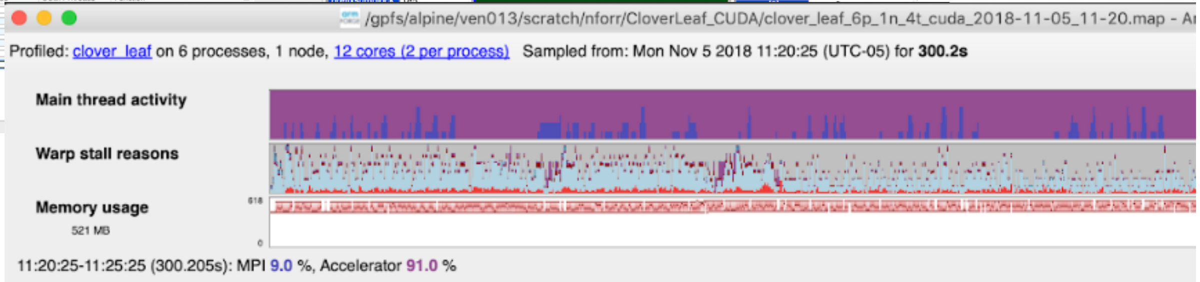
If you get GdbmiParser errors set the following environment variable

exportALLINEA_FORCE_DEBUGGER=gdb-82

Debugging and Performance Engineering for Nvidia and AMD GPUs







Linaro Forge

Python Profiling

21.0 - improved python support

- Call stacks
- Time in interpreter

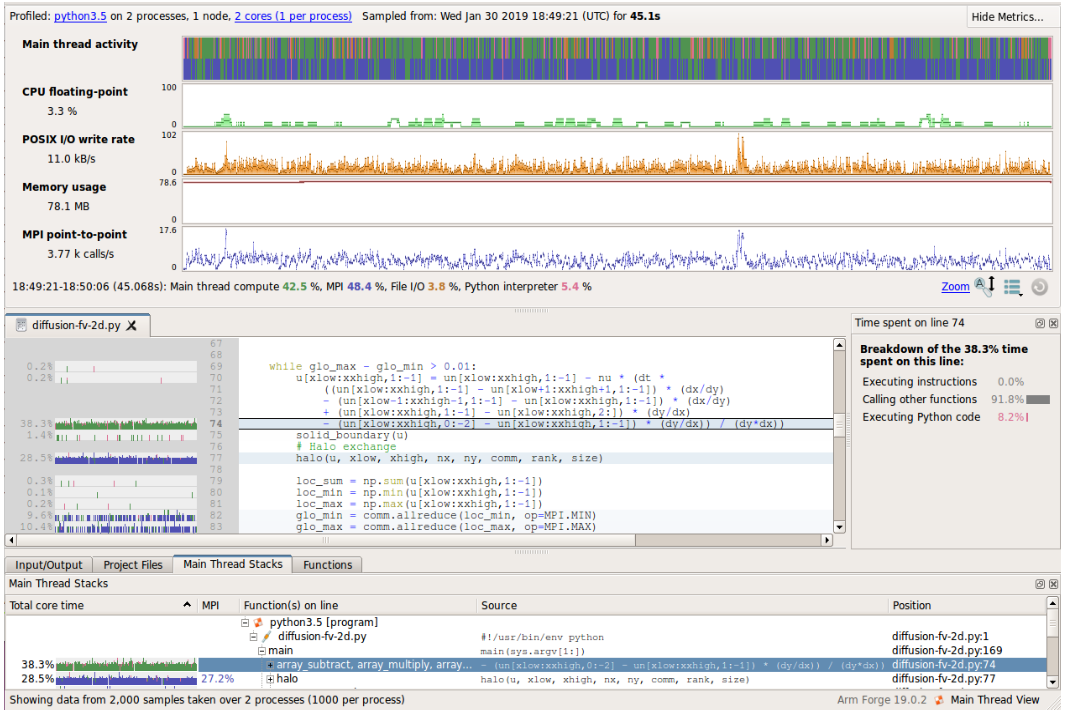
Works with MPI4PY

- Usual MAP metrics

Source code view

- Mixed language support

Note: Green as operation is on numpy array, so backed by C routine, not Python (which would be pink)




```
map --profile jsrun -n 2 python3 ./diffusion-fv-2d.py
```

Linaro Forge