

Leaders in parallel software development tools

Allinea Unified Environment

Allinea's unified tools for debugging and profiling HPC Codes

Beau Paisley
Allinea Software
bpaisley@allinea.com
720.583.0380

allinea

Leaders in parallel software development tools















































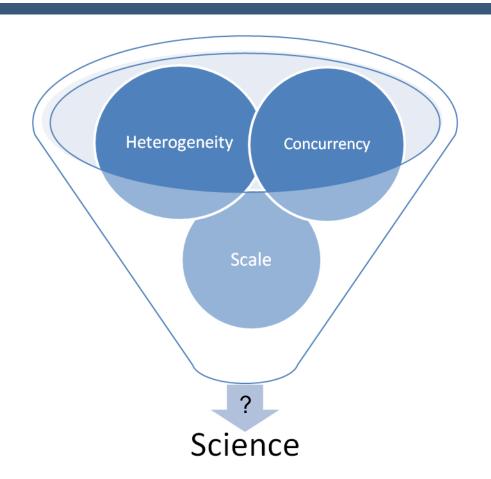








Today's Challenge



Q: What is the impact of current trends in HPC on your application?

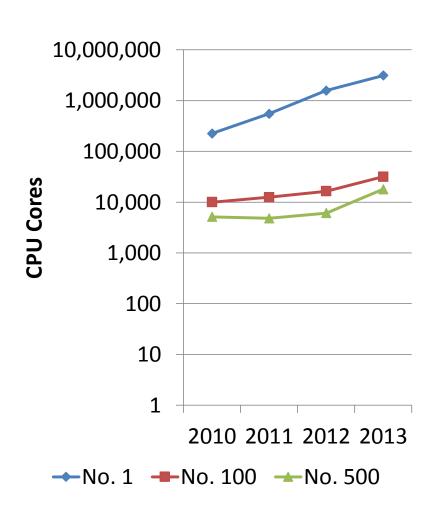
Q: How can you make your science run well on the available system?

A: Development.

Development implies both fixing problems and optimizing the computation.



Machine Size Growth



Machine sizes are exploding

Software scale grows as machines grow

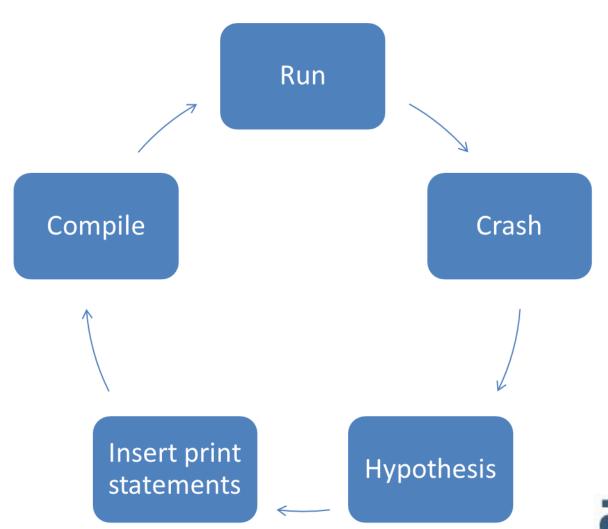


Compilers Can't do it All, ...

```
mg.f(2432): (col. 10) remark: loop was not vectorized: not inner loop.
mg.f(2431): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(993): (col. 13) remark: LOOP WAS VECTORIZED.
mq.f(992): (col. 10) remark: loop was not vectorized: not inner loop.
mq.f(991): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(243): (col. 7) remark: loop was not vectorized: existence of vector depende
mg.f(993): (col. 13) remark: LOOP WAS VECTORIZED.
mg.f(992): (col. 10) remark: loop was not vectorized: not inner loop.
mq.f(991): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(753): (col. 13) remark: loop was not vectorized: vectorization possible but
seems inefficient.
mg.f(762): (col. 13) remark: loop was not vectorized: vectorization possible but
seems inefficient.
mq.f(749): (col. 10) remark: loop was not vectorized: not inner loop.
mg.f(746): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(993): (col. 13) remark: LOOP WAS VECTORIZED.
mq.f(992): (col. 10) remark: loop was not vectorized: not inner loop.
mg.f(991): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(2255): (col. 16) remark: loop was not vectorized: existence of vector depen
dence.
mg.f(2254): (col. 13) remark: loop was not vectorized: not inner loop.
mg.f(2251): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(2433): (col. 13) remark: LOOP WAS VECTORIZED.
mg.f(2433): (col. 13) remark: loop was not vectorized: not inner loop.
mg.f(2432): (col. 10) remark: loop was not vectorized: not inner loop.
mg.f(2431): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(2433): (col. 13) remark: LOOP WAS VECTORIZED.
mg.f(2433): (col. 13) remark: loop was not vectorized: not inner loop.
mg.f(2432): (col. 10) remark: loop was not vectorized: not inner loop.
mg.f(2431): (col. 7) remark: loop was not vectorized: not inner loop.
mg.f(527): (col. 7) remark: loop was not vectorized: nonstandard loop is not a v
ectorization candidate.
mg.f(552): (col. 7) remark: loop was not vectorized: nonstandard loop is not a v
ectorization candidate.
mg.f(1150): (col. 7) remark: loop was not vectorized: loop was transformed to me
mset or memcpy.
mg.f(1150): (col. 7) remark: loop was not vectorized: loop was transformed to me
mset or memcpy.
mg.f(1645): (col. 7) remark: loop was not vectorized: loop was transformed to me
                 (7) remark: loop was not vectorized: loop was transformed to me
```

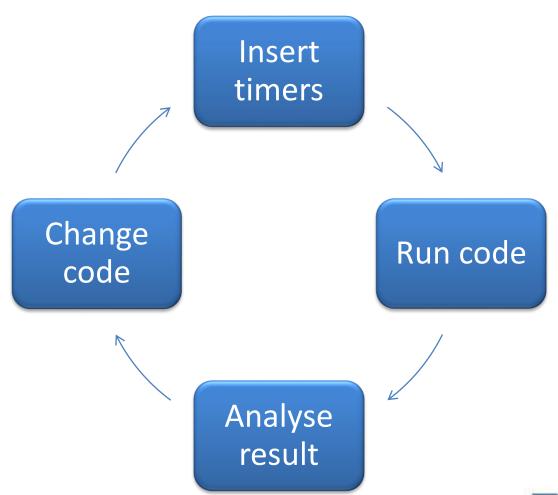


Debugging in practice...





Optimization in practice...





Some Bug Types

Bohrbug	Steady, dependable bug
Heisenbug	Vanishes when you try to debug (observe)
Mandelbug	Complexity and obscurity of the cause is so great that it appears chaotic
Schroedinbug	First occurs after someone reads the source file and deduces that it never worked, after which the program ceases to work



Allinea Unified Environment

A modern integrated environment for HPC developers

 Supporting the lifecycle of application development and improvement

Allinea DDT: Productively debug code

Allinea MAP: Enhance application performance

Allinea Performance Reports: Characterize
 Application performance

- Designed for productivity
 - Consistent easy to use tools
 - Enables effective HPC development
- Improve system usage
 - Fewer failed jobs
 - Higher application performance





An Integrated Environment





Use Allinea MAP to find a bottleneck

Flick to Allinea DDT to understand it

Compare variables, expressions, call paths

High memory usage? Use DDT to find out why

Common interface and settings files





Allinea DDT Fix software problems - fast

Graphical debugger designed for:

- C/C++, Fortran, UPC, CUDA, CUDA Fortran, OpenACC
- Multi-threaded code
- Multi-process code
- Accelerated codes
 - GPUs, Intel Xeon Phi
- Debugging 1 to 700k processes
- Slash your time to debug :
- Reproduces and triggers your bugs instantly
- Helps you to fix them as swiftly as possible
- Helps you easily understand where issues come from quickly





Allinea DDT: Debugging that Scales



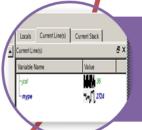
Where?

- Leaps to source automatically
- Powerful instantaneous memory debugging



How?

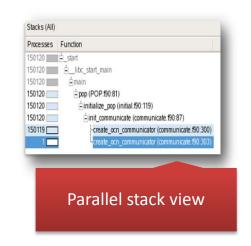
- Real-time data comparison and consolidation
- Identify outliers and unusual threads

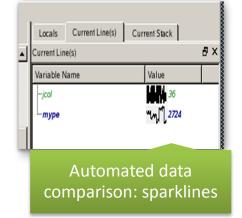


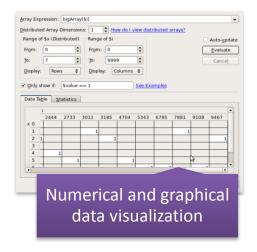
Why?

- "Smart Highlighting" of differences and changes
- Sparklines comparing data across processes

Top Features for HPC Debugging

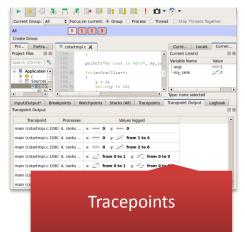














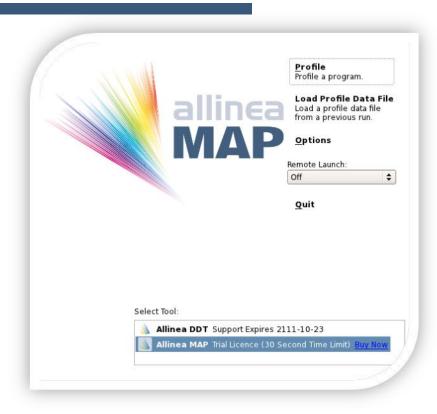
Allinea MAP Increase Application Performance

Parallel profiler designed for:

- C/C++, Fortran
- Multi-process code
 - Interdependent or independent processes
- Multi-threaded code
 - Monitor the main threads for each process
- Accelerated codes
 - GPUs, Intel Xeon Phi

Improve productivity :

- Helps you detect performance issues quickly and easily
- Tells you immediately where your time is spent in your source code
- Helps you to optimize your application efficiently





Find Performance Issues Quickly

Look at the entire application on real data sets

- Visualize the entire run at full scale, not just reduced sets
- Zoom in to explore iterations, functions and loops

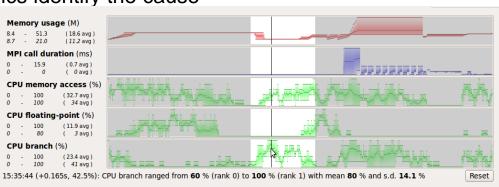
Non-Destructive Profiling

- Less than 5% overhead
- No need to instrument your code
- Small output files (10-20Mb is typical)

Understand the nature of bottlenecks

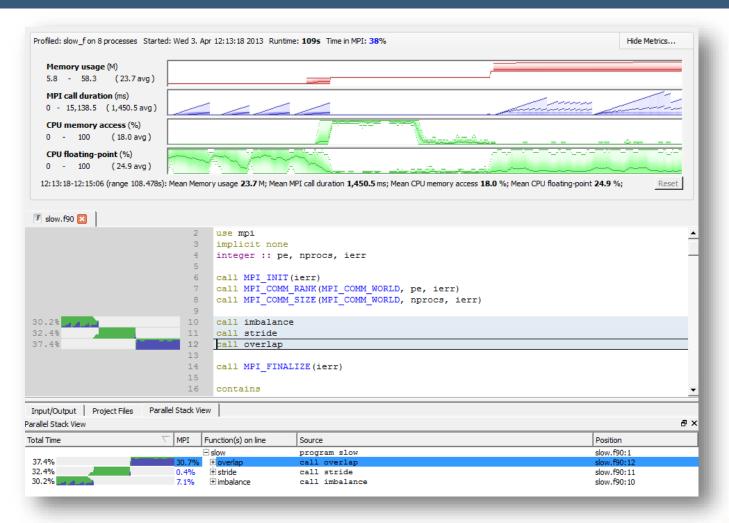
- Source code viewer pinpoints bottleneck locations
- CPU, MPI, I/Os and memory metrics identify the cause







Providing Visual Scalability



Common horizontal axis



Aggregate across all processes



Highlight imbalance visually



Always refer to source code



Allinea Performance Reports



Executable: Resources: Machine: Start time: Total time: Full path: Notes:

cp2k.popt 256 processes, 16 nodes cray-one Tue Oct 27 16:02:12 2013 951 seconds (16 minutes) /users/allinea/cp2k/exe/CRAY-XE6-gfortran-hwtopo H20 benchmark



Summary: cp2k.popt is CPU-bound in this configuration

The total wallclock time was spent as follows:



Time spent running application code. High values are usually good.

This is average; check the CPU performance section for optimization advice.

Time spent in MPI calls. High values are usually bad.

This is average; check the MPI breakdown for advice on reducing it.

Time spent in fliesystem I/O. High values are usually bad. This is negligible; there's no need to investigate I/O performance

This application run was CPU-bound. A breakdown of this time and advice for investigating further is in the CPU section below.

CPU

A breakdown of how the 56.5% total CPU time was spent:
Scalar numeric ops 27.7%

Vector numeric ops 11.3%

Memory accesses 60.9%

Other 0.0

The per-core performance is memory-bound. Use a profiler to identify time-consuming loops and check their cache performance. Little time is spent in vectorized instructions. Check the compiler's vectorization advice to see why key loops could not be vectorized.

I/U

A breakdown of how the 0.0% total I/O time was spent:

Time in reads 0.0%
Time in writes 0.0%
Estimated read rate 0 bytes/s
Estimated write rate 0 bytes/s

No time is spent in I/O operations. There's nothing to optimize here!

MPI

Of the 43.5% total time spent in MPI calls:
Time in collective calls 8.2%
Time in point-to-point calls 91.8%

Time in point-to-point calls 91.8%

Estimated collective rate 169 Mb/s

Estimated point-to-point rate 50.6 Mb/s

The point-to-point transfer rate is low. This can be caused by inefficient message sizes, such as many small messages, or by imbalanced workloads causing processes to walt. Use an MPI profiler to identify the problematic calls and ranks.

Memory

Per-process memory usage may also affect scaling:

Mean process memory usage 82.5 Mb
Peak process memory usage 89.3 Mb
Peak node memory usage 7.4%

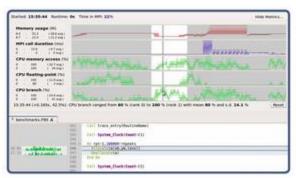
The peak node memory usage is low. You may be able to reduce the total number of CPU hours used by running with fewer MPI processes and more data on each process.

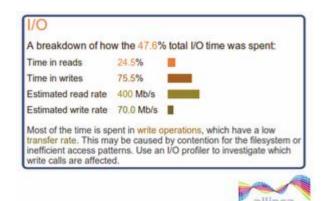
- Effortless one-touch reports
 - Add one command to your run script
 - A one-page report is generated automatically
- Characterize and understand application performance
 - With < 5% application slowdown



Top Features for HPC Code Optimization

- Allinea's tools provide extensive performance metrics, with low overhead
- Allinea's tools provide a graphical, easy-to-use presentation that is easily understood by scientists, engineers, and software developers
- Allinea MAP shows exactly which lines of source code are slow and why without modifications or instrumentation
- Allinea Performance Reports offers application level performance characterization and advice









Remote Access Clients for Mac, Windows and Linux

Easier access to distant clusters

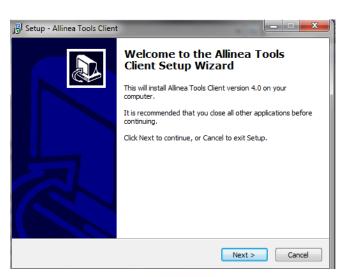
- Scalable debugging tree already cuts down network traffic
- Secure low-latency debugging and profiling clients

Extends existing remote cluster support to cover

- No shared filesystem
- Remote/local source-viewing
- Support for multi-hop SSH and OTP systems

Allinea DDT feature set available remotely

- Linux, Windows and OS/X clients
- Real native GUI no 'VNC' or 'X-forwarding' lag





What Our Users are Saying



"My group routinely debugs code at over 100,000 processes using Allinea DDT. No other debugger comes close – obviously it's a hit with users," Oak Ridge National Laboratory



"Allinea's experience and tools will make a big impact in the speed at which scientists can complete their research," NCSA Blue Waters



"Previous experiences with other profilers had left us more confused than informed. Allinea MAP is the opposite."





Leaders in parallel software development tools

Thank You

Try it out at:

http://www.allinea.com/products/trials/



