Debugging at Scale

Lindon Locks
llocks@allinea.com
• At scale debugging - from 100 cores to 250,000
  – Problems faced by developers on real systems
  – Alternative approaches to debugging and how they stack up
  – How Allinea makes debugging at scale work
• HPC tools since 2001
  – Allinea DDT – Scalable parallel debugger
  – Allinea OPT – Optimisation tool for MPI and non-MPI
  – Allinea DDTLite – Parallel debugging plugin for Microsoft Visual Studio

• Large customer base
  – Ease of use – means tools get used
  – Users debugging regularly at all scales: 1 to 100,000 cores
  – World's only Petascale debugger
Some Clients and Partners

- **Academic**
  - Over 200 universities

- **Major research centres**
  - ANL, CEA, EPCC, GENCI, IDRIS, Juelich, NERSC, ORNL

- **Aviation and Defence**
  - Airbus, AWE, BAE, Dassault, DLR, EADS

- **Energy**
  - CGG Veritas, IFP, Total

- **EDA**
  - Cadence, Intel, Synopsys

- **Climate and Weather**
  - UK Met Office, Meteo France, NOAA
• Processor counts growing rapidly

• GPUs entering HPC

• Large hybrid systems imminent

• But what happens when software doesn't work?
Problems at Scale

• Increasing job sizes leads to unanticipated errors
  – Regular bugs
    • Data issues from larger data sets – eg. garbage in..., overflow
    • Logic issues and control flow
  – Increasing probability of independent random error
    • Memory errors/exhaustion – “random” bugs!
    • System problems – MPI and operating system
  – Pushing coded boundaries
    • Algorithmic (performance)
    • Hard-wired limits (“magic numbers”)
  – Unknown unknowns
    • ....
• Improved coding standards

  – Unit tests, assertions and consistency checks
    • Good practice – but tend to be single-process checks
    • Parallel checks also valid and good practice

  – Only checks for things you predict when developed
    • Coverage is rarely perfect
      – Unexpected problems – particularly random/system issues – often missed
    • Debugger still required

  – Combines well with debuggers
    • Find why a failure occurs not just a pass/fail
Strategies for bug fixing II

• Logging – printf and write
  – The oldest debugger still in active use
    • Tried and tested - as easy as “hello world”
    • If you have good intuition into the problem
      – Edit code, insert print, recompile and re-run
    • **Slow and iterative**
  – Use to log exceptions, progress or state
    • Post-mortem analysis only
      – Hard to establish real causal order of output of multiple processes
      – Output can be lost by process termination
    • Rapid growth in log output size
    • **Unscalable**
Strategies for bug fixing III

- Reproduce at a smaller scale
  - Attempt to make problem happen on fewer nodes
    - Often requires reduced data set – the large one may not fit
      - Didn't you already try the code at small scale?
      - Smaller data set may not trigger the problem
    - Does the bug even exist on smaller problems?
    - Is it a system issue – eg. an MPI problem?
  - Is probability stacking up against you?
    - **Example:** 1 in 10,000 independent probability of error?
    - Unlikely to spot on smaller runs – without many many runs
    - But near guaranteed to see it on a 10,000 core run
  - What can a parallel debugger do to help?
    - Debug at the scale of the problem. **Now.**
• Many benefits to graphical parallel debuggers
  – Large feature sets for common bugs
  – Richness of user interface and real control of processes

• Historically **all** parallel debuggers hit scale problems
  – Bottleneck at the frontend: Direct GUI → nodes architectures
    • Linear performance in number of processes
  – Human factors limit – mouse fatigue and brain overload

• Are tools ready for the task?
  – Allinea DDT has changed the game
DDT in a nutshell

- **Scalar features**
  - Advanced C++ and STL
  - Fortran 90, 95 and 2003: modules, allocatable data, pointers, derived types
  - Memory debugging

- **Multithreading & OpenMP features**
  - Step, breakpoint etc. one or all threads

- **MPI features**
  - Easy to manage groups
  - Control processes by groups
  - Compare data
  - Visualise message queues
Scalable Process Control

- Parallel Stack View
  - Find rogue processes quickly
  - Identify classes of process behaviour
  - Rapid grouping of processes

- Control Processes by Groups
  - Set breakpoints, step, play, stop for groups
  - Scalable groups view: compact group display
Handling Regular Bugs

- Immediate stop on crash
  - Segmentation fault, or other memory problems
  - Abort, exit, error handlers
  - CUDA errors

- Scalable handling of error messages

- Leaps to the problem
  - Source code highlighted
  - Affected processes shown
  - Process stacks displayed clearly in parallel
Finding the cause

• Full class/structure browsing
  – Local variables and current line(s)
    • Show variables relevant to current position
    • Drag in the source code to see more
  – C, C++, F90: object members, static members, derived types

• Automatic comparison and change detection
  – Scalable and fast
Finding rogue processes

- Easy to find where the differences are...
  - Cross process comparison of data
    - Fetches values from every process, compares and then groups by value
    - Summary of NaN, Inf and statistics
  - Easy to spot rogues

- Use to group processes
  - Define a process group and control en-masse
Large Array Support

- Browse arrays
  - 1, 2, 3, … dimensions
  - Table view
- Filtering
  - Look for an outlying value
- Export
  - Save to a spreadsheet
- View arrays from multiple processes
  - Search terabytes for rogue data: in parallel with [v3.0]
• Find memory leaks

• Or stop on read/write beyond end of array:
DDT is delivering Petascale debugging today

- Collaborations with ORNL on Jaguar Cray XT and CEA
- Logarithmic performance
- Many operations now faster at 220,000 than previously at 1,000 cores
- \(~1/10\text{th of a second}\) to step and gather all stacks at 220,000 cores
Summary

• Debuggers are recognised as the right tools to fix bugs quickly: other methods have limited success, and major issues at scale

• Debugging interfaces must scale to help the user understand what is happening

• Allinea DDT scales in performance and interface – breaking all records and making problems manageable

• See Allinea at Supercomputing 2010: Booth 2305
Questions?