Simplifying complex software development environments at scale

David Lecomber
CTO Allinea
Coming up...

• Introduction
  – Allinea and tools in HPC

• DDT
  – Brief overview, scalability focus and update

• OPT
  – Brief overview and update

• Questions
Parallel Software is Complicated

• Multithreaded, multiprocess code
  – The usual issues: bugs, speed ...
  – ... now add communication, synchronization, race conditions, deadlock, scalability ....
  – ... unpredictability of behaviour between systems, and within same system

• Now more complex.. several architectures
  – Hybrid Cell and Opteron

• Hybrid GPU and x86_64
  – Homogeneous heavyweight-kernel clusters
  – Homogeneous lightweight-kernel clusters
  – Large SMP machines
  – Desktop SMP via multicore

• No clear winner yet
  – Development nightmare!
  – Can you do it without tools?
Three Challenges Today

• Languages and compilers
  – What do I use currently?
  – What will I use on my next system?

• Making a code that works right
  – How do I debug today?
  – How will I debug my next system?

• Making a code that works fast
  – How do I optimize today?
  – How will I optimize my next system?

• Lots of choice for the language – but few for the rest
Allinea Software

• HPC tools company since 2001

• Core products
  – **DDT** - Debugger for MPI, threaded/OpenMP and scalar applications
  
    – **OPT** - Optimizing and profiling tool for MPI and non-MPI applications

• New product
  – **DDTLite** - Plugin for Microsoft Visual Studio 2008
    • Adds parallel and multi-threaded components to user interface
    • Real parallel debugging for Windows!
    • Released September 22nd 2008
High Profile Clients (extract)

- **Grids**
  - SHARCNet, ICHEC, North West Grid

- **Universities**
  - FZ Jülich, Karlsruhe, Dresden, HLRS Stuttgart, LRZ Munich
  - Oxford, Cambridge, Warwick, Manchester
  - Vanderbilt, TACC, Michigan, Oregon, Indiana, Penn State, Wisconsin, Alberta

- **Aerospace research**
  - DLR, EADS CCR, CIRA, MBDA, CERFACS, Dassault, BAe Systems

- **Commercial research**
  - Airbus, Fujitsu, CGG, CGG Veritas, Total, IFP, OHM, AVL, MTEM, Intel

- **Research centres**
  - CEA, NERSC, IDRIS, BSC, ONERA, RAL, HLRS, CASPUR, CINECA, NERSC, LLNL

- **Weather/Climate**
  - Met Office, BGS, Proudman, Ifremer, Mercator
DDT - Distributed Debugging Tool

- A mature, powerful and highly intuitive tool
  - Traditional focus has been HPC
- Cross-platform support
  - Linux, Solaris (Sparc, x86-64), CLE, AIX
  - GNU, Absoft, IBM, Intel, PGI, PathScale, Sun compilers
  - Blue Gene, Cell, x86-64, ia64, Power, UltraSparc, NEC
- Across all MPI and OpenMP implementations
  - From low end to high end
- Support for all scheduling systems
  - SGE, PBS, LSF, MOAB, ...
  - Flexible, powerful, easy to use queue submission
DDT: Basic Principles

- Sophisticated GUI helps the user to control parallel execution and helps to find and focus on potential problems
- User controls actions by groups..
  - Set breakpoints, lock step, align stacks etc
  - But can focus in on individual threads / processes when necessary
- Create groups both
  - Manually: select processes via drag and drop
  - Automatically: by process stack, values etc
Features for every model

• Scalar features
  – Advanced C++ support including STL, namespaces, virtual functions and templates
  – Advanced Fortran 90, 95 and 2003 support including modules, allocatable data, pointers and derived types

• Multithreading & OpenMP features
  – Perform actions individually or collectively

• MPI features
  – Control processes
  – individually or by groups
  – Visualize message queues
Memory Debugging
• Cross process/thread comparison
• Visualize multidimensional data
  – 3D
  – From distributed debugger
• Advanced user defined data display
  – Program DDT to display your data using your software!
• Single interface for more complex architectures
  – eg. IBM Cell BE – combined view of all stacks of PPE and SPEs
  – Combined control including step, play and pause
  – DDT ready to meet challenges of new solutions: modern GUI will continue to be extended
Scalable Debugging

• **Control Processes by Groups**
  – Set breakpoints, step, play, stop etc. using user-defined groups of processes
  – Scalable process groups view

• **Parallel Stack View**
  – Finds rogue processes faster
  – Identifies classes of process behaviour easily
  – Allows rapid grouping of processes

• **Parallel Variable View**
  – Find rogue data faster
  – Integrated with process groups
Current Scalability

• What has been achieved in last 12 months?

• Scalable GUI
  – For the first time debug 5,000 with same ease as 100
  – At a glance full stack and status of all processes

• 10x improvement in scale limits
  – Iterative improvement has brought benefit
  – Debugging 5,000 processes is comfortable
  – Regular users at 4096 cores
  – Test rig emulating 16,000 cores at native speed

• High end platforms
  – BlueGene/P support added to list Q3/08
  – Cray XT4, XT5 users at scale
  – Ranger at TACC – Infiniband Sun Constellation cluster
Latest results with DDT

- **Good for all of most of today's systems**
  - Highly parallel architecture has served us well
    - All process debugging done on parallel nodes
    - GUI interprets and displays results
    - Some system architectures better than others..
  - GUI already scales for presentation
    - Permits new tools – eg. plugin MPI checkers

- **Tomorrow: Need to beat linear performance**
  - “Infinitely scalable” performance via multi-level network
Optimizing in a Parallel Universe…

- **Traditional tracers**
  - Timelines:
    - Good for watching messages and memory accesses to pick out problems visually
  - But not easily scalable!
- **Can log everything but…**
  - Vast quantities of data are generated
  - Is it really necessary?
  - Analysis becomes an expert task
- **Is MPI the only game in parallel computing?**
  - Of course not...
  - Cell, GPU, desktop multi-core
  - New programming models, new challenges
...Keep It Simple

- **Focus is the key!**
  - Too much visual information can be confusing
  - Good parallel tools should simplify things
  - Tools should target the areas which cause problems
  - Directing the user towards the problem points...

- **Allinea OPT**
  - A 'top-down' focused approach:
    - See the “big picture” first – call graph
    - Drill down successively for more information..
  - Don’t drown users (or system!) in too much data
    - Mixture of sampling and selective tracing
  - Supports most cluster flavours, and IBM/Sony Cell
    - New: IBM BlueGene/P support
OPT – Making Optimization Easier
Optimizing Hybrids

- How do we optimize hybrids?
  - Can traditional products extend to GPU, or Cell?

- Need to show core behaviour
  - It's where the most computation happens..
  - No simple “gprof” support yet for most hybrids

- Need to measure data transfer costs and offloaded op times

- OPT now available for Cell
  - Shows SPU and PPU actual processor usage per function
Summary

• Architectural complexity is already here

• Scale is coming
  – Top 500 June 2008 – 30 systems with > 10,000 cores

• Debugging and optimizing at scale
  – Some problems appear only at scale
  – Need scalable debugging performance
  – Need a scalable GUI: the brain is the bottleneck

• Yet, we must continue to innovate at a lower scale
  – Most problems are solved at lower scale – even on the larger systems
  – How many systems are < 10,000 cores?
  – Persistent and reverse debugging MPI and scalar codes in DDT 2.4

• Our goal
  – Make picking up a tool easier, more instinctive, than printf
  – Bugs get fixed faster with debuggers
Thank you