Blue Waters Software Tools Overview

Gregory Bauer, Senior Technical Program Manager
gbauer@illinois.edu
Blue Waters Software Tools

Three general categories

• Productivity
• Debugging
• Performance
Blue Waters OS Support

- SLES 11 SP3*
  - 3.0 Linux kernel
  - GNU libc 2.11.3
- GPU Support
  - CUDA 9.1
- Lustre parallel file system
IEEE: The Top Programming Languages 2019

<table>
<thead>
<tr>
<th>Rank</th>
<th>Language</th>
<th>Type</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Python</td>
<td>☮️</td>
<td>100.0</td>
</tr>
<tr>
<td>2</td>
<td>Java</td>
<td>☮️</td>
<td>96.3</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>☮️</td>
<td>94.4</td>
</tr>
<tr>
<td>4</td>
<td>C++</td>
<td>☮️</td>
<td>87.5</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>☮️</td>
<td>81.5</td>
</tr>
<tr>
<td>6</td>
<td>JavaScript</td>
<td>☮️</td>
<td>79.4</td>
</tr>
<tr>
<td>7</td>
<td>C#</td>
<td>☮️</td>
<td>74.5</td>
</tr>
<tr>
<td>8</td>
<td>Matlab</td>
<td>☮️</td>
<td>70.6</td>
</tr>
<tr>
<td>9</td>
<td>Swift</td>
<td>☮️</td>
<td>Apple</td>
</tr>
<tr>
<td>10</td>
<td>Go</td>
<td>☮️</td>
<td></td>
</tr>
</tbody>
</table>

Programming languages for data-Intensive HPC applications

https://doi.org/10.1016/j.parco.2019.102584
Software Environment

Languages
- Fortran/CAR (OpenACC)
- C (OpenACC)
- C++ (OpenACC)
- Python
- UPC

Compilers
- Cray Compiling Environment (CCE)
- GNU
- Intel

Programming Models
- Distributed Memory (Cray MPT)
  • MPI
  • SHMEM
- Shared Memory
  • OpenMP 3.0

IO Libraries
- NetCDF
- HDF5
- ADIOS

Tools
- Environment setup
- Modules
- Debugging Support Tools
- • Fast Track Debugger (CCE w/ DDT)
  • Abnormal Termination Processing

Optimized Scientific Libraries
- LAPACK
- ScaLAPACK
- BLAS (libgoto)
- Iterative Refinement Toolkit

Debuggers
- Cray Performance Monitoring and Analysis Tool
- Allinea DDT
- Igdb

Resource Manager
- PGAS & Global View
  • UPC (CCE)
  • CAF (CCE)

Shared Memory
- • OpenMP 3.0

Adaptive
- STAT
- Cray Comparative Debugger
- Data Transfer

Visualization
- VisIt
- Paraview

Traditional
- Charm++

PGAS & Global View
- • OpenMP 3.0

Traditional
- Cray Linux Environment (CLE)/SUSE Linux

Cray developed
Under development
3rd party packaging
Licensed ISV SW
NCSA supported
Cray added value to 3rd party
Productivity Tools

- Compilers (C/C++/Fortran)
  - Cray, GNU, Intel, PGI
    https://bluewaters.ncsa.illinois.edu/compiling
- Interfaces
  - MPI, OpenMP (more later)
- Frameworks
  - Charm++ (more later)
  - PETSc
    https://bluewaters.ncsa.illinois.edu/petsc
- IDEs
  - Eclipse

Libraries

- https://bluewaters.ncsa.illinois.edu/libraries
  - Linear Algebra
    - CPU Cray LibSCI, Intel MKL
    - GPU Cray LibSCI_ACC, cuBLAS, MAGMA
  - FFTs
    - CPU Cray LibSci, Intel MKL
    - GPU cuFFT
  - IO
    https://bluewaters.ncsa.illinois.edu/io-libraries
    - HDF5
    - IOBUF
Blue Waters Programming Environment

- The modules utility is used to select a particular programming environment.
  - The module command is used to swap, remove and add.
    - Swap from Cray to GNU: `% module swap PrgEnv-cray PrgEnv-gnu`
    - Add CUDA environment: `% module add cudatoolkit`
    - See [https://bluewaters.ncsa.illinois.edu/modules](https://bluewaters.ncsa.illinois.edu/modules) and module man page for more information.
- Compiler wrappers make it easy to link in libraries added via modules
  - `cc`, `CC` and `ftn` call appropriate compiler (`craycc`, `gcc`, `icc`, etc) and link in MPI, OpenMP, CUDA libraries.
  - Control linkage model: static (default) or dynamic (`-shared`) to compilers.
  - See `cc`, `CC` and `ftn` man pages for more information.
Python on Blue Waters - bwpy

- Custom python installation to address several issues:
  - the need to have consistency of modules within python distribution,
  - address not optimally configured modules, and
  - the impact of walking the PYTHONPATH and module loading on a large, parallel filesystem.
- More later in the webinar.
Shifter – Containers for HPC

- Runs a chroot environment similar to Docker
- Like Docker, allows the user to substitute the entire runtime environment except the kernel
- Focus on providing Docker while maintaining full HPC security
- Can use images from Docker hub as well as a local registry or flattened files.
- Reduces Lustre metadata burden
- Multiple teams have and are using Shifter
- Max Belkin will provide more information in a later presentation.
Workflow Software

- Scheduler.x (simple)
  - A simple MPI code that takes in a list of work and executes it.
  - Used in EGM project (more later)
  - [https://github.com/ncsa/Scheduler](https://github.com/ncsa/Scheduler)
  - [https://bluewaters.ncsa.illinois.edu/job-bundling#schedulerdotx](https://bluewaters.ncsa.illinois.edu/job-bundling#schedulerdotx)
Workflow Software

- Swift/Parsl (complex)
  - Supports directed acyclic graphs (DAG) structured workflows.
  - Swift – Java, custom scripting language.
    - [https://bluewaters.ncsa.illinois.edu/swift](https://bluewaters.ncsa.illinois.edu/swift)
  - Parsl – Python based. Similar in structure to Swift.
    - [https://parsl-project.org/](https://parsl-project.org/)
  - Used by EarthDEM on Blue Waters.
- Webinars - [https://bluewaters.ncsa.illinois.edu/webinars/workflows](https://bluewaters.ncsa.illinois.edu/webinars/workflows)
Machine Learning (more later)

• Packages available in bwpy
  • Pytorch, Tensorflow, Scikit-learn
  • Not tracking latest version available.
• Vendor libraries
  • MKL Deep Neural Network Library (DNNL)
  • NVIDIA cuDNN and NCCL
Debugging (more later)

- ATP – Abnormal Termination Processing
  - Provides stack traces, outlier processes, viewable .dot files.
  - Provides a subset of all core files.
  - Linked into code by default.
  - Enable at runtime.
  
  https://bluewaters.ncsa.illinois.edu/atp

- GDB
  - For traditional use cases.

https://docs.nersc.gov/programming/performance-debugging-tools/stat_atp/
Debugging (more later)

- ARM Forge
- Scalable, graphical, interactive, full features, unattended, replay, ...
- formerly known as Allinea DDT

https://bluewaters.ncsa.illinois.edu/ddt
Performance Analysis

- CrayPAT
  - Automatic or manual code instrumentation.
  - Command line or GUI.
  - Profile code w.r.t. CPU performance, communication, IO, etc.

https://bluewaters.ncsa.illinois.edu/cpmat
Performance Analysis

- System Monitoring
  - Staff can see overall performance over time.
  - Can generate job report for initial overview.
  - Can highlight imbalances and inefficiencies.
- Contact us.
Performance Analysis (IO)

- **Darshan**
  - Library that aggregates IO performance for MPI-based applications.
  - Linked in by default but module can be unloaded.
  - Can use log file to generate report.