Python based applications on Red Storm

Porting a Python based application to the Lightweight Kernel

May 10, 2007

John Greenfield

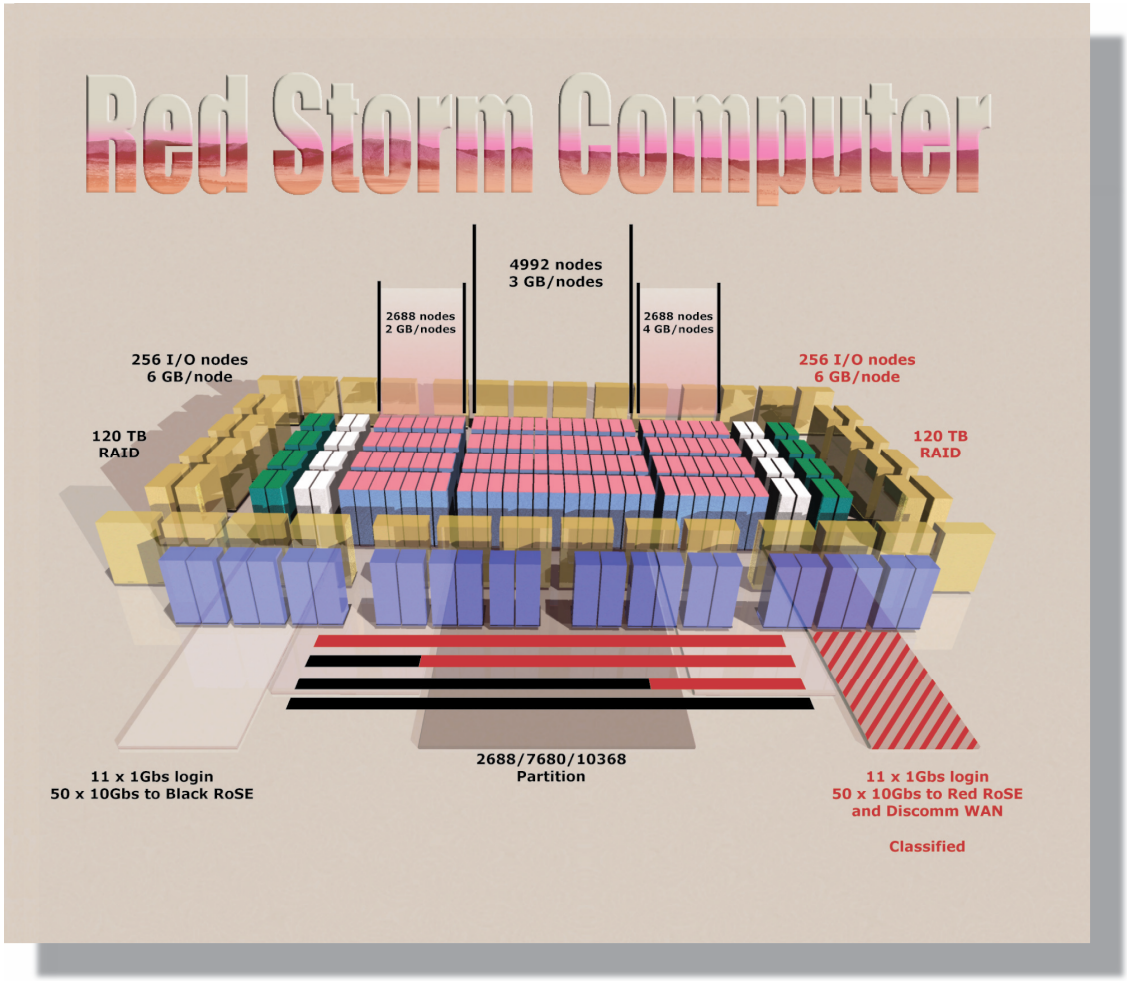
Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Agenda

- Red Storm
- User needs - Why Python
- Difficulties
  - No Dynamic Libraries
  - Cross-Compiling Issues
  - Other Difficulties
  - Parallel Performance
- Conclusions
- Future Work

Sandia National Laboratories

Red Storm



Sandia National Laboratories

User needs

- Post-processing applications
  - Verification and Validation
  - Scaling
  - Quantification of Uncertainty
  - Error checking
  - Optimization and parameter studies
- Quick Easy transfer from other machines
  - Have codes that work, need them on Red Storm
  - Often these are in Python

Data Services Toolkit (DSTK)

- Our original motivating application
- Toolkit for post-processing Exodus files
  - Subsetting and selection
  - Algebra operations on data
  - Parallel capability
- Python interface to a set of tools in C
- Uses pyMPI, numerics Python modules
- Written as Python module (dstk.py)

Sandia National Laboratories

Difficulties

- No Dynamic Libraries
- Cross-Compiling Issues
- Other Difficulties
- Parallel Performance

Sandia National Laboratories

No Dynamic Libraries

- Compile Statically
- Replace Python loader
  - Look for static libraries instead
- Modules and libraries used increase size
- For small calculations, size not an issue
- If most of module used, size penalty minimal
- Need to recompile if compilers or libraries change.

Sandia National Laboratories

Cross Compiling Issues

- Need to handle yod as well as compiling on service nodes for compute nodes.
- Need to provide return codes that yod doesn't
  - Wrapper to pass result codes via file.
- Modify make to use yod as launcher.
- Need service node version of python to build third-party modules.

Other difficulties

- Handle system variable settings
  - NGROUP_MAX and TMP_MAX set to 0
  - Used by Python for array size and loop settings
  - Undef to get Python defaults
- Avoid explicit large file support
  - Causes use of 64-bit file IO, which has bugs
  - Disabling will use 32-bit IO functions

Sandia National Laboratories

Parallel Performance

- Python loads modules from disk at run time
- Loading from disk doesn't scale for diskless nodes.
- Revise python loader to load via MPI bcast
  - Load single rank from disk
  - Broadcast to rest of nodes
  - Requires all nodes load same modules at start

Sandia National Laboratories

Step-by-step build

- Build and install Python module builder
  - Service node
- Build and install basic Python
  - For compute node
- Build Third-party modules
- Add modules to Python static link list
- Rebuild Python with final module list

Sandia National Laboratories

Conclusion

- Performance
  - Speed as good as or better than dynamic linking
  - Size not much bigger, especially for limited number of modules.
- Extension to other codes
  - Static linking and parallel efficiency easy to translate
  - Other difficulties more code specific
    - Look out for system variable setting assumptions

Sandia National Laboratories

Future Work

- Porting Python-based simulation code
  - Dynamically selects modules
  - Size from excess modules a concern
  - Working on automatic re-linking to add modules dynamically
- Reimplementing DSTK functionality in ParaView
  - Should be available late this year
  - Need to port ParaView to Red Storm
    - Planned project for next year

Acknowledgements

- Thanks to colleagues
  - Rena Haynes and Jim Holton for DSTK development
  - Sandia Visualization teams for advice and support
  - Red Storm System teams for assistance
- Funding provided by ASC DVS program.

Sandia National Laboratories