Expanding an HPC Cluster to Support the Computational Demands of Digital Pathology[1,2]

C. Campbell, N. Mecca, T. Duong, I. Obeid and J. Picone
The Neural Engineering Data Consortium, Temple University
{christopher.campbell, nmecca, thuc.duong, iobeid, picone}@temple.edu

The goal of this work was to design a low-cost computing facility that can support the development of an open source digital pathology corpus containing 1M images [1]. A single image from a clinical-grade digital pathology scanner can range in size from hundreds of megabytes to five gigabytes. A 1M image database requires over a petabyte (PB) of disk space. To do meaningful work in this problem space requires a significant allocation of computing resources. The improvements and expansions to our HPC (high-performance computing) cluster, known as Neuronix [2], required to support working with digital pathology fall into two broad categories: computation and storage. To handle the increased computational burden and increase job throughput, we are using Slurm [3] as our scheduler and resource manager. For storage, we have designed and implemented a multi-layer filesystem architecture to distribute a filesystem across multiple machines. These enhancements, which are entirely based on open source software, have extended the capabilities of our cluster and increased its cost-effectiveness.

Slurm has numerous features that allow it to generalize to a number of different scenarios. Among the most notable is its support for GPU (graphics processing unit) scheduling. GPUs can offer a tremendous performance increase in machine learning applications [4] and Slurm's built-in mechanisms for handling them was a key factor in making this choice. Slurm has a general resource (GRES) mechanism that can be used to configure and enable support for resources beyond the ones provided by the traditional HPC scheduler (e.g. memory, wall-clock time), and GPUs are among the GRES types that can be supported by Slurm [5]. In addition to being able to track resources, Slurm does strict enforcement of resource allocation. This becomes very important as the computational demands of the jobs increase, so that they have all the resources they need, and that they don't take resources from other jobs. It is a common practice among GPU-enabled frameworks to query the CUDA runtime library/drivers and iterate over the list of GPUs, attempting to establish a context on all of them. Slurm is able to affect the hardware discovery process of these jobs, which enables a number of these jobs to run alongside each other, even if the GPUs are in exclusive-process mode.

To store large quantities of digital pathology slides, we developed a robust, extensible distributed storage solution. We utilized a number of open source tools to create a single filesystem, which can be mounted by any machine on the network. At the lowest layer of abstraction are the hard drives, which were split into 4 60-disk chassis, using 8TB drives. To support these disks, we have two server units, each equipped with Intel Xeon CPUs and 128GB of RAM. At the filesystem level, we have implemented a multi-layer solution that: (1) connects the disks together into a single filesystem/mountpoint using the ZFS (Zettabyte File System) [6], and (2) connects filesystems on multiple machines together to form a single mountpoint using Gluster [7].

ZFS, initially developed by Sun Microsystems, provides disk-level awareness and a filesystem which takes advantage of that awareness to provide fault tolerance. At the filesystem level, ZFS protects against data corruption and the infamous RAID write-hole bug by implementing a journaling scheme (the ZFS intent log, or ZIL) and copy-on-write functionality. Each machine (1 controller + 2 disk chassis) has its own

1. Research reported in this publication was most recently supported by the National Human Genome Research Institute of the National Institutes of Health under award number U01HG008468. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

2. This material is also based in part upon work supported by the National Science Foundation under Grant No. CNS-1726188. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

separate ZFS filesystem. Gluster, essentially a meta-filesystem, takes each of these, and provides the means to connect them together over the network and using distributed (similar to RAID 0 but without striping individual files), and mirrored (similar to RAID 1) configurations [8].

By implementing these improvements, it has been possible to expand the storage and computational power of the Neuronix cluster arbitrarily to support the most computationally-intensive endeavors by scaling horizontally. We have greatly improved the scalability of the cluster while maintaining its excellent price/performance ratio [1].

REFERENCES

[1] D. Houser, G. Shadhin, R. Anstotz, C. Campbell, I. Obeid, J. Picone, T. Farkas, Y. Persidsky and N. Jhala, "The Temple University Hospital Digital Pathology Corpus," *IEEE Signal Processing in Medicine and Biology Symposium*, 2018, p. 1.

[2] C. Campbell, N. Mecca, I. Obeid, and J. Picone, "The Neuronix HPC Cluster: Improving Cluster Management Using Free and Open Source Software Tools," *IEEE Signal Processing in Medicine and Biology Symposium*, 2017, p. 1.

[3] A. B. Yoo, M. A. Jette, and M. Grondona, "SLURM: Simple Linux Utility for Resource Management," in *Job Scheduling Strategies for Parallel Processing*, 2003, pp. 44–60.

[4] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[5] "Generic Resource (GRES) Scheduling." [Online]. Available: *https://slurm.schedmd.com/gres.html*.

[6] J. Bonwick, M. Ahrens, V. Henson, M. Maybee, and M. Shellenbaum, "The Zettabyte File System," in *Proceedings of the 2nd Usenix Conference on File and Storage Technologies*, 2003, pp. 1–13.

[7] "What is Gluster?" [Online]. Available: *https://docs.gluster.org/en/v3/Administrator Guide/GlusterFS Introduction/*.

[8] B. Depardon, G. Le Mahec, and C. Seguin, "Analysis of Six Distributed File Systems," Institut National de Recherche en Informatique et en Automatique (INRIA), Lyon, France, 2013. *https://hal.inria.fr/hal-00789086/document*.

Expanding an HPC Cluster to Support the Computational Demands of Digital Pathology

C. Campbell, N. Mecca, T. Duong, I. Obeid and J. Picone

The Neural Engineering Data Consortium, Temple University

NEURAL ENGINEERING DATA CONSORTIUM
www.nedcdata.org

College of Engineering
Temple University

Abstract

- NEDC is currently developing an open source corpus of 1M digital pathology images as part of its NSF-funded Major Research Instrumentation grant.
- This project required a cost-effective approach to hosting and backing up 1 Petabyte of online storage.
- Purchasing cloud-based storage typically costs $0.01 – 0.02/GB/month, while our on-premises solution costs $0.06/GB. This is a much lower cost when prorated over a three or five-year period.
- To implement this storage platform, we purchased two storage nodes, each connected to 960TB of raw disk space to house the pathology corpus, as well as a 96TB storage node to provide additional storage space for our computing environment.
- Because our network is heterogenous, we created a Storage Area Network (SAN) using GlusterFS, with individual storage nodes using ZFS. Although the storage nodes in the SAN run Linux, the storage pool can be used by Windows machines.
- To handle the management of compute resources and improve job throughput, we are using SLURM as our job submission platform.

Introduction

- Digital pathology presents unique computational demands due to the high resolution images required (e.g., 5K x 5K pixels consuming 5GB in disk space).
- Much of the software which supports clinical and research use cases of digital pathology has been written specifically for Windows.
- However, many of the open source HPC, machine learning and storage toolkits and platforms are designed to work on Linux/UNIX systems, so a storage platform which could work equally well with Windows and Linux systems is required.

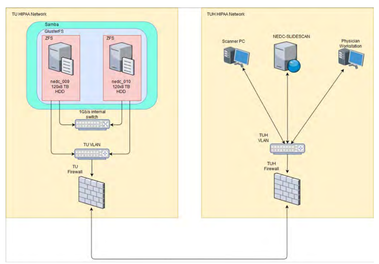

Fig. 1: A heterogeneous storage solution

- This system must be able to function in a hospital setting, where the system has to span multiple HIPAA-secured networks and firewalls, and reside in multiple physical locations.
- The Samba interface can be configured to authenticate against local credentials or can use an organization's existing LDAP database, allowing physicians to have a single sign-on.

Cross-platform Petabyte-scale Storage

- At the hardware level, each storage node is attached via a SAS 8644 cable to two external chassis which can hold 60x 8TB HDD, for a total of 480 TB/chassis.
- ZFS was then used to provide the backend filesystem, as well as the RAID implementation. Each storage node has a separate ZFS filesystem.
- To create the SAN, GlusterFS was used. Volumes are distributed (combines the available disk space of all nodes) or replicated (data is written to all nodes).
- Two separate pools in the same SAN were created: the primary pool and the backup pool. Either pool can be mounted from either machine.

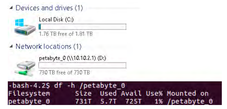

Fig. 2: The primary pool mounted on both Windows and Linux machines.

- A Gluster pool is mounted from a Windows machine using a Samba share.

Managing Compute Resources

- Due to the computationally-demanding deep learning algorithms we use to process the data, a powerful and flexible job scheduler is needed.
- Support for scheduling GPUs as a separate item is a necessity, since many of our machines have multiple GPUs on a single motherboard.
- A typical compute node has 384 GB of RAM and 4x NVIDIA Tesla P40 GPUs (24GB DDR5 RAM.
- A single-threaded machine learning job needs to use multiple GPUs, so the scheduler needs to be able to effectively timeshare jobs that use different numbers of GPUs.
- SLURM allows users to request resources according to a number of parameters (e.g. number of CPU cores or number of GPUs), ensuring that jobs do not experience resource bottlenecks.

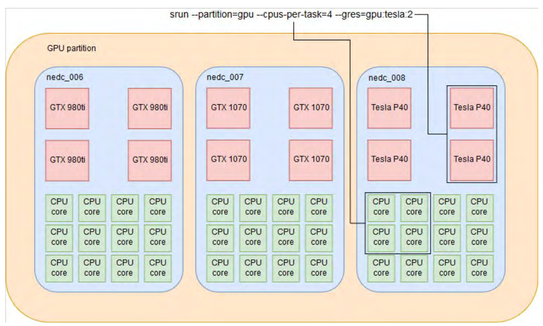

Fig. 3: Example of resource selection in SLURM on the Neuronix cluster

- In the example above, a user can, with a single command, select a partition (queue), the number of CPU cores allocated for it (see Fig. 4).
- With job accounting, long-term usage statistics can be collected to find bottlenecks.

Job Scheduling

- In addition to handling compute resources, the other important function performed by SLURM is scheduling jobs.
- In cluster computing settings, it is very common for multiple users, or groups of users, to run jobs on the same resources, so it is important that the job scheduler be able to schedule jobs to run in a fair and reasonable manner.
- SLURM has multiple mechanisms to support different scheduling scenarios, including grouping users into accounts and organizations, with the ability to set job priority for a specific user, group, or organization.
- Beyond a user's association, SLURM can determine job priority and schedule according to any number of parameters, including how long the job has been queued, the amount of resources requested, which queue the job was submitted to, and the user's history of resource usage.

The Neuronix Use Case

- Using SLURM as our job manager, we can exert a very fine degree of control over the physical resources that a job will have allocated to it.

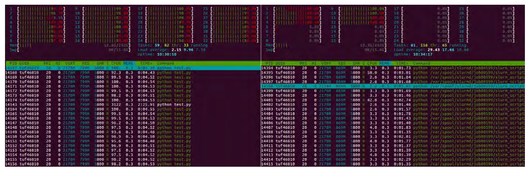

Fig. 4: Running a job via ssh (left), and via SLURM (right)

- Neuronix compute nodes are grouped by function as SLURM partitions. Within a particular partition, nodes with specific features can be chosen, or nodes can be selected from a general resource (GRES) pool (e.g. GPUs).

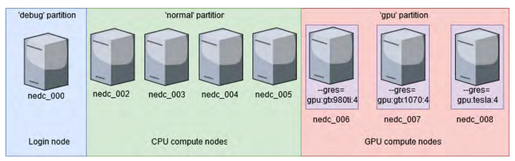

Fig. 5: A map of the Neuronix compute nodes

- Using the ZFS/GlusterFS storage pool, the total storage available to our HPC cluster is 96TB, with all 7 dedicated compute nodes able to access the pool without needing any of their own local storage.
- A modest file server (2x Intel® Xeon® E5-2620 v4, 128GB RAM) is able to support the 96TB volume and handle requests from the entire cluster.
- Cost effectiveness: the two petabyte machines were purchased for $90k in total, which yields $0.045/GB of raw disk space, and $0.06/GB of usable filesystem space. This compares very favorably to cloud-based solutions, especially at this scale where prices can exceed $10K/month for high-availability storage.

Scalability and Future Expansions

- Both solutions discussed were designed with horizontal scalability in mind, which comes with the advantage of not having to decommission old hardware after an expansion.
- When the storage nodes are fully loaded, identical machines can be configured the same way with ZFS, and added to the Gluster pool.
- When a node is added to the storage pool, Gluster can re-balance it to distribute the storage load.
- The Neuronix cluster's storage space has been expanded to 96TB using the same strategy as the petabyte machines, so the total available storage can be expanded indefinitely.
- We are investigating switching from 1Gb/s Ethernet to Infiniband, which provides extremely high network throughput with low latency.
- SLURM can be configured to work with cloud-based compute nodes, and custom hooks can be written to allocate and release resources on platforms like AWS or Azure as jobs start and stop, allowing for the integration of techniques like bursting.

Summary

- A new system architecture was developed using open-source tools that can scale to meet the needs of our digital pathology research.
- ZFS, Gluster and Samba were used to create a petabyte-scale storage platform that is both highly scalable and cross-platform.
- SLURM allows compute resources to be demarcated for specific types of jobs, which ensures that jobs requiring more computing power are able to request the appropriate resources and nodes.
- Both the computing and storage systems developed can be easily implemented from low-cost commodity components to accommodate almost any type of computing or workload-requirement.

Acknowledgements

- Research reported in this publication was most recently supported by the National Human Genome Research Institute of the National Institutes of Health under award number U01HG008468. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.
- This material is also based in part upon work supported by the National Science Foundation under Grant No. CNS-1726188. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.