Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Maximizing the performance of scientific data transfer by optimizing the interface between parallel file systems and advanced research networks

Nicholas Mills [a,*], F. Alex Feltus [b], Walter B. Ligon III [a]

[a] Holcombe Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, United States

[b] Department of Genetics and Biochemistry, Clemson University, Clemson, SC, United States

HIGHLIGHTS

- A variety of configurations for performing scientific data transfers is evaluated.
- Optimal data transfer parameters are identified for the test network.
- Some parameters are generalized with an equation involving properties of the network.

ARTICLE INFO

Article history:
Received 19 September 2016
Received in revised form
24 March 2017
Accepted 16 April 2017
Available online xxxx

Keywords:
Software defined networking
High throughput computing
DNA sequence
Parallel data transfer
Parallel file system
Data intensive science

ABSTRACT

The large amount of time spent transferring experimental data in fields such as genomics is hampering the ability of scientists to generate new knowledge. Often, computer hardware is capable of faster transfers but sub-optimal transfer software and configurations are limiting performance. This work seeks to serve as a guide to identifying the optimal configuration for performing genomics data transfers. A wide variety of tests narrow in on the optimal data transfer parameters for parallel data streaming across Internet2 and between two CloudLab clusters loading real genomics data onto a parallel file system. The best throughput was found to occur with a configuration using GridFTP with at least 5 parallel TCP streams with a 16 MiB TCP socket buffer size to transfer to/from 4–8 BeeGFS parallel file system nodes connected by InfiniBand.

1. Introduction

Solving scientific problems on a high-performance computing (HPC) cluster will happen faster by taking full advantage of specialized infrastructure such as parallel file systems and advanced software-defined networks. The first step in a scientific workflow is the transfer of input data from a source repository onto the compute nodes in the HPC cluster. When the amount of data is small enough it is possible to store datasets within the local institution for quick retrieval. However, for certain domains ranging from genomics to social analytics the amount of data is becoming too large to store locally. In such cases the data must be optimally transferred from an often geographically distant central repository.

In genomics, our primary area of interest, the velocity of data accumulation due to high-throughput DNA sequencing should not be underestimated, as data accumulation is accelerating into the Exascale. As of this writing there are six quadrillion base pairs in the sequence read archive database at NCBI [1] with each A,T,G,C stored as at least two bytes. These data represent publicly mineable DNA datasets across the Tree of Life from viruses to rice to humans to elephants. New powerful genetic datasets such as The Cancer Genome Atlas [2] contain deep sequencing from tumors of over 11,000 patients; sequences of over 2500 human genomes from 26 populations have been determined (The 1000 Genomes Project [3]); genome sequences have been produced for 3000 rice varieties from 89 countries (The 3000 Rice Genomes Project [4]). These raw datasets are but a few examples that aggregate into petabytes with no end of growth in sight. In fact, if DNA sequencing technology continues to improve in terms of resolution and cost, one could view DNA sequencing as an Internet of Things application with associated computational challenges across the DNA data life cycle [5].

* Corresponding author.
E-mail addresses: nlmills@g.clemson.edu (N. Mills), ffeltus@clemson.edu (F.A. Feltus), walt@clemson.edu (W.B. Ligon III).

http://dx.doi.org/10.1016/j.future.2017.04.030
0167-739X/

Domain scientists who are not experts in computing often resort to transferring these large datasets using FTP over the commodity Internet at frustratingly slow speeds. Even if a fast network is available, large transfers can take on the order of several days to complete if the correct transfer techniques are not used. The HPC and Big Data communities have developed several tools for fast data transfers but these tools are often not utilized. Several factors contribute to the lack of utilization by scientists of the proper tools. First, awareness of the proper tools in the community is lacking to the point where researchers result to familiar but technically inferior tools such as FTP. Old habits are hard to break. Second, a lack of adequate documentation means even when researchers are pointed to the right tools they are lost in a maze of configuration options. Manuals give an overview of various parameters but fail to explain what parameter values are suitable for high-performance data transfers.

An end-to-end data transfer involves at least three major components: a high-performance storage system, a high-performance network, and the software to tie it all together. The various research communities have all examined optimized performance parameters for their respective components. Unfortunately, our experience has been that independently optimizing a single component leads to slower performance in the system as a whole. This work attempts to fill the gaps by suggesting optimized transfer parameters based on experimentally measured end-to-end transfer performance. Where possible the experimentally-determined parameters are supported with appropriate theory.

1.1. Storage component

When considering storage requirements it is important to remember that data transfer is only the first step in an HPC workflow. Presumably there will later be a significant amount of computation performed on the data in parallel. While technologies such as large single-node flash storage arrays may provide the highest raw storage throughput during a transfer, those arrays will quickly become a bottleneck during the computation phase of the workflow as multiple compute nodes compete for access to the centralized storage component. Instead, a parallel file system (PFS) provides a good compromise between raw storage bandwidth and parallel processing scalability.

A PFS is a special case of a clustered file system, where the data files are stored on multiple server nodes. Storing a file on n nodes where the nodes can be accessed simultaneously theoretically allows for a speedup of n times over the single-node case. In practice, the achievable speedup is often much less than n and depends on both the parallel file system implementation and the access patterns of the application performing I/O. It is therefore advantageous to evaluate multiple parallel PFSs to find the one most suited to a particular workload.

In this paper some of the more popular PFSs are evaluated. Among the file systems compared are BeeGFS [6], Ceph [7], GlusterFS [8], and OrangeFS [9]. BeeGFS is later used for file transfer tests once it is determined to have the best performance in a benchmark. Lustre is another popular PFS that was not evaluated because it is not supported for the Ubuntu operating system [10].

1.2. Network component

Access to a fast network with plenty of excess capacity is essential for high performance data transfers. The Science DMZ model [11] is well suited to data transfers because of its support for the long-lived elephant flows typical of large transfers. In particular the lack of firewalls in the model prevents slowdown caused by packet processing overhead. The network infrastructure for our experiments is provided as a part of the CloudLab environment [12–14].

CloudLab is a platform for exploring cloud architectures and evaluating design choices that exercise hardware and software capabilities. CloudLab is a great benefit to researchers because it allows them to quickly and easily test different modern hardware and software configurations without the usual troubles associated with re-installing the operating system and re-configuring the network [15]. CloudLab allows the allocation of *bare metal* machines with no virtualization overhead. Thus, CloudLab is a realistic experimental environment for moving real datasets within and between sites before moving algorithms and procedures into production.

A major capability of CloudLab is the ability to connect clusters together over Internet2 using software-defined networking. Our experiments use two of these clusters: the Apt cluster in the University of Utah's Downtown Data Center in Salt Lake City, and the Clemson cluster of Clemson University in Anderson, South Carolina [16].

Communication between the Apt and Clemson CloudLab clusters occurs over the Internet2 network using the Advanced Layer-2 Service (AL2S) [17]. AL2S allows nodes in the two clusters to transparently communicate as if they were connected to the same switch. That is, they appear to be on the same IP subnetwork. The only discernible difference is the relatively high communication latency of 26 ms when compared to a more typical latency of 180 μs.

Configuring a new path over AL2S would normally involve multiple technical and administrative hurdles. The experimenter would need to contact the appropriate network administrator and convince them to set up a new path between sites. The network administrator would need to configure the new path with Internet2 via the GlobalNOC [18]. The delay involved in this process could be significant. For a production network that may run for several years or more such a delay may not matter. But for a temporary experimental network intended to run on the order of weeks or less the large initial setup delay can be significant.

A major advantage of CloudLab for multi-site experiments is that the CloudLab software configures new links over AL2S automatically at the beginning of each experiment. In most cases this process is seamless and requires only a couple minutes of waiting. A researcher need only specify the two sites to be connected and the CloudLab infrastructure handles the rest.

1.3. Software component

Two software tools drive experiments. The first tool, XDD [19], is used to perform benchmarks of parallel file systems. As a benchmark tool XDD makes use of its knowledge of disks to achieve maximum performance by queueing multiple large I/O requests simultaneously. In our experiments XDD uses multiple threads to access different regions of the same file in parallel.

The second tool, GridFTP [20], is a well-known application for performing large data transfers. GridFTP enhances the original FTP protocol with extensions designed to support faster data transfers. It also has a wide variety of networking configuration options. GridFTP uses a client–server model where the server provides access to the host file system, and the client can either push or pull data relative to the server.

2. Materials and methods

2.1. Test dataset

The test dataset used for file transfer experiments is composed of DNA sequencing data in Sequence Read Archive format. There are 345 files in the dataset that range in size from 643 MB to 11 GB. The median file size is 2.4 GB. Over a quarter of the files are less than 1 GB, and less than 11% are greater than 4.5 GB.

Please cite this article in press as: N. Mills, et al., Maximizing the performance of scientific data transfer by optimizing the interface between parallel file systems and advanced research networks, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.04.030