

Accessing HDF5 Files via SSHFS (FUSE)

MuQun Yang
CHoonghwan Lee
Scott Wegner
The HDF Group
November 2010

When working with HDF5 data files, it is often necessary to access only a small subset of the data. We have found in such situations that FUSE and SSHFS may provide an efficient alternative to NFS when accessing subsets of remotely stored HDF5 data.

1 Introduction

The HDF5 file format is frequently used for storing large pools of scientific data in which files are often many gigabytes in size. Typically these large files are generated by a specialized supercomputer and transferred to a scientist's local machine for analyzing. If only a small portion of the data needs to be analyzed, then much time is wasted transferring the HDF5 file over the network. A better solution would be to transfer only the necessary portions of the file.

Using an NFS [1] mount point, users can access HDF5 files remotely as if they were stored locally, and then use tools such as h5copy to transfer only the subset of data that is needed. NFS is not an ideal tool, though, because it requires administrator privileges to set up and access mount points. NFS presents a convenient solution for smart remote file manipulations, but it is often unavailable to users because of its administrator requirements.

FUSE [2], which stands for "filesystem in userspace", is a standard module to the Linux kernel that provides an API for implementing user-defined filesystems. With the module loaded, any user can mount a user-defined filesystem without the need for administrator privileges. Many filesystems including SSHFS [3] have been implemented by using FUSE. SSHFS allows users to mount an SSH access point as a filesystem, allowing convenient access to remote files. SSHFS is also efficient in that when a file is opened on the mounted filesystem, only minimal data is transferred. SSHFS sends partial file data via an SSH protocol only when a user reads or writes portions of the file.

Together with FUSE and SSHFS, users can access files on systems to which they have remote access and manipulate them with their local toolset. We conducted a series of experiments using FUSE and SSHFS in conjunction with HDF5 utilities, such as h5copy [4], and found FUSE and SSHFS to be simple to set up while still realizing the benefits of efficient remote file access.

2 Experiments

We conducted two experiments to verify the usefulness of SSHFS and analyzed the efficiency in typical situations. All experiments were done using HDF-EOS5 data files, ranging in size between 64 KB to 1.22 GB. Data files were stored on a remote, faster Linux server, and accessed using SSHFS and SCP from a local, slower Linux server on the network. To mitigate inconsistencies in network response time, each point in the experimental data represents the average of several identical trials.

2.1 Remote File Subsetting

In the first experiment, we staged the scenario in which a large HDF5 file exists on the remote server, but we are only interested in a single dataset. Thus, we need to copy the dataset to the local machine for analysis. We approached this problem in two ways. Using the standard approach, we transferred the entire file using SCP, and then used h5copy to make a new HDF5 file containing only the selected dataset. We assume that the elapsed time of transferring the entire file using SCP is close to downloading the file from the remote machine to the local machine. Alternatively, we accessed the same file on an SSHFS mount and used h5copy to make a local HDF5 file with only the pertinent dataset. We measured the time taken to complete each set of operations and computed the ratio of the time taken for SSHFS and h5copy to complete the task and the time taken to transfer the entire file via SCP. The ratios are plotted against the size of the selected dataset in comparison to the overall HDF5 file as shown in Figure 1. The experiment was run with 32 different HDF5 files. These file sizes range from 64 KB to 650 MB. The ratio of the size of a selected dataset to the size of the corresponding HDF5 file ranges from less than 1 percent to 50 percent.

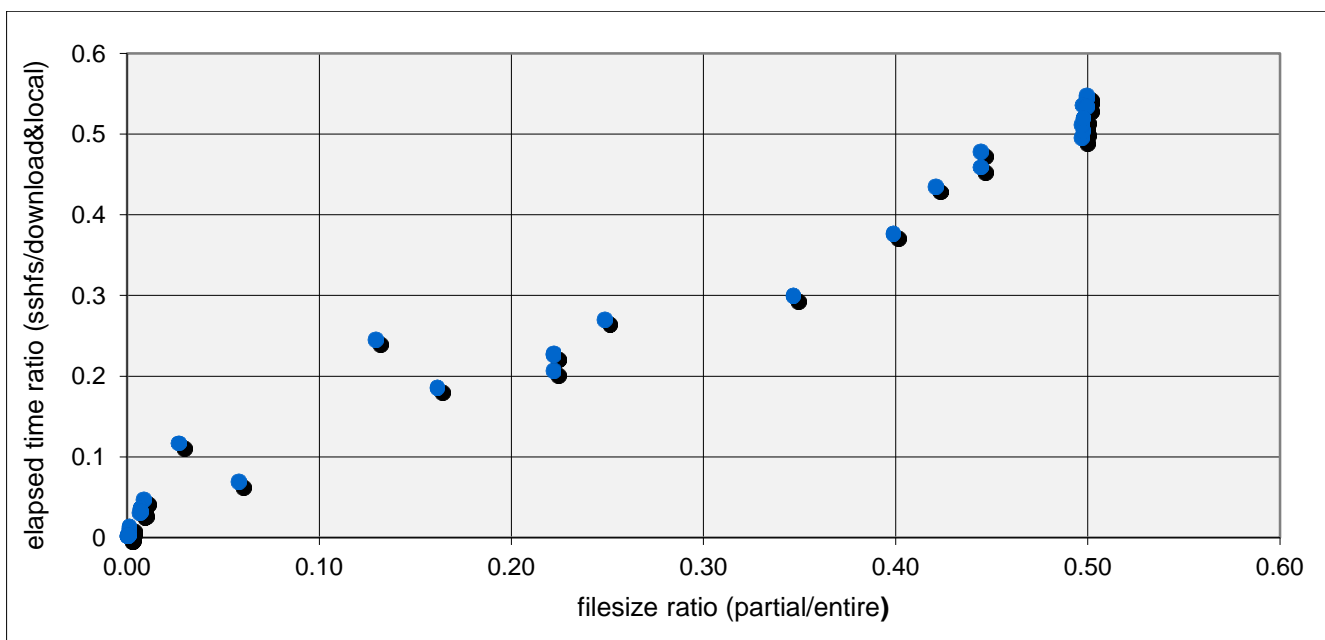


Figure 1: The distribution of ratios of elapsed time of transferring selected datasets of HDF5 files via SSHFS (FUSE) to the elapsed time using SCP for 32 HDF5 files. The X-axis represents the ratio of the size of the selected dataset to the size of the whole file. The Y-axis represents the ratio of the elapsed time of transferring the selected datasets via SSHFS to the elapsed time using SCP.

In all experiments, using SSHFS for subsetting was more efficient than copying the whole file. In fact, as the ratio of the size of the selected dataset to the size of the whole file increases, the ratio of the

elapsed time of transferring the selected dataset of an HDF5 file via SSHFS to the elapsed time of transferring the entire file via SCP and then accessing the selected dataset locally also increases, at the same pace. For example, when the ratio of the size of the selected dataset to the size of the whole file is about 0.5, the ratio of elapsed time of transferring the selected dataset via SSHFS to the elapsed time of transferring the whole file using SCP and then accessing the selected dataset locally is also about 0.5. This indicates that by using SSHFS only the selected dataset is transferred from the remote machine to the local machine. The overhead of using SSHFS is also relatively small.

2.2 Subsetting with Hyperslab Selection

In our second experiment, we compared the efficiency of using SSHFS in cases when only a portion of the dataset is needed. The experiment was run with three different HDF5 files, ranging in file sizes from 546 MB to 1.22 GB. For this experiment, we used SSHFS in three different ways. Firstly, we copied the entire HDF5 file via SSHFS to the local machine in order to access the file directly. Secondly, we used the approach from the previous experiment, using SSHFS with h5copy to transfer only data in the relevant dataset. And finally, we used SSHFS and copied only the portion of the dataset that was needed. In this final approach, we created a modified version of the h5copy utility that utilizes the HDF5 hyperslab selection APIs. We tested each method on the three HDF5 files and compared those elapsed times to that of the elapsed time for copying the file with SCP and subsetting it on the local Linux machine.

Table 1 shows the sizes of the files, the selected datasets and the selected hyperslabs used in this experiment. Table 2 shows the ratios of elapsed time via SSHFS to the elapsed time via SCP and local access. The elapsed time via SSHFS is almost the same as the elapsed time via SCP and local access when copying the entire file. The elapsed time via SSHFS reduces dramatically when using SSHFS to access the selected datasets and the hyperslabs. Figure 2 shows the comparisons of elapsed time ratios via size ratios for the entire file, one HDF5 dataset and one hyperslab for all three files.

This experiment confirms that SSHFS transfers only the selected dataset or the hyperslab from the remote machine to the local machine. The overhead of using SSHFS is relatively small. One may note that the ratio of the elapsed time of transferring only the selected dataset via SSHFS to the elapsed time of transferring the whole file and then accessing the selected dataset locally is even smaller than the ratio of the size of the selected dataset to the entire file as shown in Figure 2. This difference occurs because, in order to have local access of the selected dataset, it requires both transferring the whole file from the remote machine to the local machine and then copying the selected dataset out at the local machine. Since the local machine is relatively slow, the elapsed time of copying a big dataset becomes significant. However, the elapsed time of copying the significantly smaller hyperslab locally has little effect on the ratio of elapsed time for accessing one hyperslab via SSHFS to the elapsed time via transferring the whole file via SCP and locally accessing the hyperslab.

Table 1: Sizes of the whole files, the selected datasets and the selected hyperslabs

Data Unit	File 1 (MB)	File 2 (MB)	File 3 (MB)
Whole file	546	934	1257
One dataset	313	535	1024
One hyperslab	39	45	64

Table 2: Ratios of elapsed time via SSHFS to elapsed time via SCP

Data Unit	Time ratio SSHFS/SCP the whole file and local access		
	File 1	File 2	File 3
Whole file	0.99	1	1
One dataset	0.46	0.47	0.62
One hyperslab	0.08	0.05	0.05

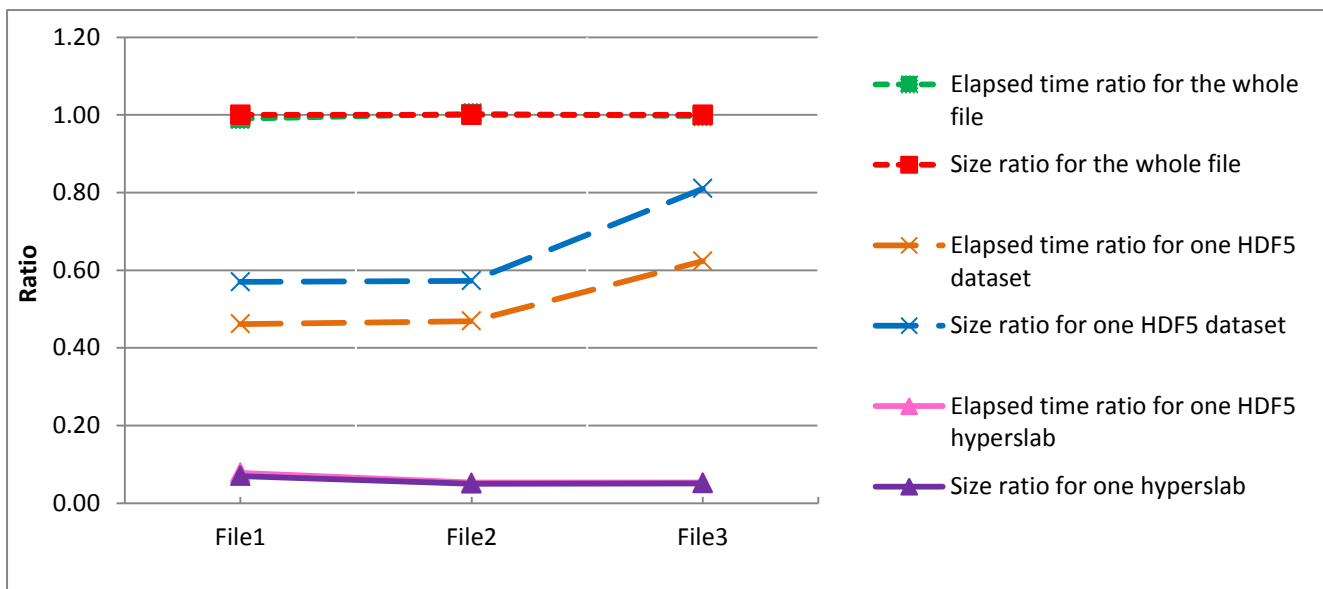


Figure 2: Comparisons of elapsed time ratios via size ratios for the whole file, one HDF5 dataset and one hyperslab. The elapsed time ratio is the ratio of the elapsed time via SSHFS (FUSE) to the elapsed time ratio via SCP and then locally accessing the whole file, one HDF5 dataset and one hyperslab. The size ratio is the ratio of the size of the whole file, one HDF5 dataset and one hyperslab to the whole file.

3 Conclusions

These experiments suggest that SSHFS can be used as a drop-in replacement for NFS when users seek to access partial HDF5 files from a remote location. SSHFS is convenient because it uses FUSE as a backend and therefore does not require administrator access. It attributes comparable efficiency as NFS for HDF5 file subsetting.

Though the breadth of these experiments is limited, it is clear from our results that dataset and hyperslab selection is beneficial for remote access and partial data transfer of large HDF5 files. One should also be aware that the emphasis of this report is to conceptually demonstrate a possible alternative solution to accessing remote data efficiently. Many factors that may affect the performance, such as the hardware of local and remote machines, are only minimally considered in this report, suggesting that perhaps under many circumstances SSHFS (FUSE) may not replace NFS or SCP. A detailed discussion is out of the scope of this report.

4 References

1. Wikipedia. "Network File System," http://en.wikipedia.org/wiki/Network_file_system.
2. SourceForge. "FUSE: Filesystem in Userspace," <http://fuse.sourceforge.net/>.
3. Sourceforge. "SSH Filesystem," <http://fuse.sourceforge.net/sshfs.html>.
4. The HDF Group. "Software Using HDF5," http://www.hdfgroup.org/products/hdf5_tools/.

5 Acknowledgments

The authors would like to thank Mr. Dan Kahn at Science Systems and Applications, Inc. (SSAI) who suggested that we investigate the performance of accessing HDF5 data via SSHFS (FUSE). We also thank Dr. Mike Folk at The HDF Group and Mr. Dan Kahn for reviewing the document. This work was supported under NASA grant NNX08A077A. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NASA.