Wonderful : A Terrific Application and Fascinating Paper

Your N. Here
Your Institution

Second Name
Second Institution

Abstract

Your Abstract Text Goes Here. Just a few facts. Whet our appetites.

1 Introduction

The gap between hard disk drives (HDDs) performance and CPUs computing capabilities, also known as the I/O gap problem, has been growing constantly in the last years and becomes particularly serious if we consider high-end computing clusters (HEC) having hundred of thousands or even millions of cores.

Distributed parallel file systems (DPFSs) such as Lustre and GPFS try to bridge this gap by striping the data across multiple servers and providing multiple parallel data paths to increase the aggregate I/O bandwidth. The abstract device I/O interface (ADIO) driver in ROMIO [5], an implementation of the MPI-IO library, enables file access optimizations, based on two-phases I/O and data sieving, that adapt I/O patterns generated by scientific applications to the underlying DPFSs [6].

Unfortunately, as Carns et al. have pointed out in their study [3], the majority of applications they analyzed still use inefficient POSIX I/O interface to access the data. It has also been demonstrated [4] that using POSIX I/O to access non-contiguous regions of files, brings to extremely poor performance in the case of networked file systems such as the parallel virtual file system (PVFS2). This is due to the high volume of requests going through the network and, of course, because of the intrinsic nature of HDDs that predilige sequential access patterns.

At the moment there is no available solution to fix non-optimal I/O patterns generated by scientific applications but re-write them (? to be checked...ATC USENIX....!). The Linux kernel provides programmers with the capability to communicate access pattern information to the file system through the system call `posix_fadvise` [1]. The file system can use this information to improve caching performance, for example, by pre-fetching data that will be required soon in the future or by disabling read-ahead in the case of random read patterns. The problem is that `posix_fadvise` is barely used and has intrinsics limitations that discourage its employment in reality.

In this paper we propose to use `posix_fadvise` to fix access patterns at run-time. We generate advice for the file system on behalf of applications using what we call Advice Manager (AM). I/O activity is monitored using an interposing library we wrote and passed to the AM that generates advice using the `posix_fadvise` system call. The advantage with our approach (is this the advantage over all other possible solutions out there??!) is that we can generate advice for applications that do not use them but can take benefit from them. We accomplish this in full asynchronism, with very low overhead, and absolutely no modification of the original application.

We demonstrate that our approach is effective to improve the storage bandwidth, reduce the number of I/O requests and the running time of a data analysis framework, widely used by physicists at CERN, called ROOT [2].

The reminder of this paper is organized as follows.

2 Related work

3 This Section has SubSections

3.1 First SubSection

3.2 New Subsection

3.3 How to Build Your Paper

3.4 Last SubSection

4 Acknowledgments

5 Availability

It's great when this section says that MyWonderfulApp is free software, available via anonymous FTP from

```
ftp.site.dom/pub/myname/Wonderful
```

Also, it's even greater when you can write that information is also available on the Wonderful homepage at

```
http://www.site.dom/~myname/SWIG
```

Now we get serious and fill in those references. Remember you will have to run latex twice on the document in order to resolve those cite tags you met earlier. This is where they get resolved. We've preserved some real ones in addition to the template-speak. After the bibliography you are DONE.

References

[1] posix_fadvise.

[2] ROOT, A Data Analysis Framework.

[3] CARNS, P., HARMS, K., ALLCOCK, W., BACON, C., LANG, S., LATHAM, R., AND ROSS, R. Understanding and improving computational science storage access through continuous characterization. In *Proc. of the 27th IEEE Conference on Mass Storage Systems and Technologies (MSST)* (2011), pp. 1–14.

[4] CHING, A., CHOUDHARY, A., KENG LIAO, W., AND PUNDIT, N. Evaluating i/o characteristics and methods for storing structured scientific data. In *Proc. of the International Parallel and Distributed Symposium* (2006).

[5] THAKUR, R., GROPP, W., AND LUSK, E. An abstract-device interface for implementing portable parallel-i/o interfaces. In *Proc. of the 6th IEEE International Symposium on The Frontiers of Massively Paralle Computation* (1996), IEEE Computer Society Press, pp. 180–187.

[6] THAKUR, R., GROPP, W., AND LUSK, E. Data sieving and collective i/o in romio. In *Proc. of the 7th IEEE International Symposium on The Frontiers of Massively Paralle Computation* (Washington, DC, USA, 1999), FRONTIERS '99, IEEE Computer Society, pp. 182–.