Parallel Virtual Shared Memory with PVFS2 and Amazon EC2

Bryan Lewis

Chicago RUG, August, 2010 (and revised October, 2010)

We discuss a technique that uses RAM across clusters to rapidly manipulate large data sets with general-purpose software like R and MATLAB.

Our approach is especially well-suited to large, I/O bound problems that benefit from a global view into the data across the cluster.

Amazon EC2 Virtual Clusters

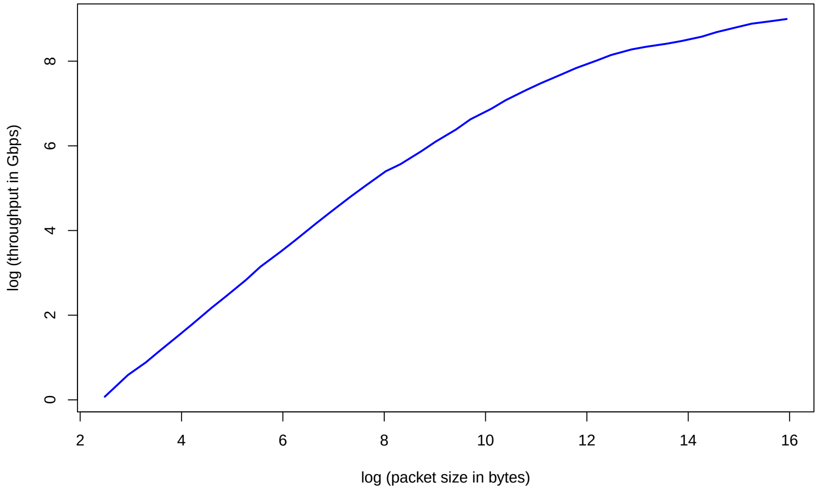
The discussed tools work with any modern GNU/Linux cluster. We illustrate them with the new Amazon Elastic Compute Cloud (EC2) virtual high-performance computing (HPC) cluster.

EC2 virtual HPC clusters are remarkable computing instruments. Each node is equipped with:

- 8 high-performance CPU cores
- 23 GB RAM
- 2 TB fast scratch storage space
- Unlimited additional Elastic Block Store (EBS) storage
- Guaranteed inter-node 10 Gbps bandwidth!

Clusters may be assembled, re-sized and torn down at will, as they are entirely virtualized.

Amazon HPC NetPIPE Throughput: Impressive!



PVFS2

PVFS2 (parallel virtual file system, version 2) is a scalable, high-performance parallel file system for Linux clusters developed at Clemson and ANL. Features include:

- Native drivers for high-performance I/O fabrics
- Mounts like a network file system or use with MPI-IO
- Emphasis on speed and scalability over features (no locking, no writable mmap)
- Very easy to set up and deploy.

Parallel file systems allow simultaneous access to the file system by multiple cluster nodes. PVFS2 scales by aggregating network and I/O bandwidth across the cluster.

Pvshm

Pvshm is a file system module for the Linux kernel that converts address space operations into read and write operations on files living in other file systems.

Pvshm provides functions to applications for management of cache consistency across the cluster. Automatic cache consistency is not enforced for performance reasons.

We use pvshm to allow applications like R to memory-map files living in PVFS2.

R and the bigmemory package

R is a powerful software environment for data analysis and statistical and scientific computing.

The *bigmemory* package defines a large matrix class for R (arrays up to about 10^{15} elements). Related packages provide various analysis, statistical and linear algebra functions.

Bigmemory can back large matrices with memory-mapped files, enabling R users to easily manipulate matrices much larger than available RAM.

EC2, PVFS2, pvshm, R, and bigmemory

Our example is simple, but represents a larger class of interesting examples.

We compute the column sums of a matrix with 5 million rows and 800 columns (about 30 GB – larger than the RAM available on a single EC2 HPC instance).

The example is I/O bound–it takes a lot longer to access the data than to compute the result.

We compare timings for sums computed sequentially and in parallel and when backed by a file on an EC2 EBS drive and backed by PVFS2/pvshm RAM across three EC2 HPC compute instances.

Wall-clock times (in seconds)

File system	EC2 Instances	Sequential	3 Threads
Amazon EBS (ext3)	1	620	758
pvshm/PVFS2 RAM	3	110	33

The parallel EBS time is larger than the sequential time because the problem is I/O bound–the extra CPU cores spend most of their time waiting for data and just incur extra overhead. PVFS2 and RAM offer sufficient bandwidth to put the extra cores to use.

The pvshm result yields about a 20$\times$ performance increase over EBS for approximately three times the cost.

Comparison and commentary

Our example is a simple one that is easy to split up–a column sum only requires data in its column.

The example could just as well be computed with a map/reduce approach like Hadoop, MPI scatter/gather, and others.

However, similar examples are not always so easily split up. Consider, for example, a problem that requires bootstrapping across all the columns of a matrix. Such a problem is hard to fit into a map/reduce framework but works well with our approach.

Summary

PVFS2 and pvshm provide virtual pools of RAM across a cluster that general-purpose programs like R may use to easily manipulate large shared data in parallel.

The approach is good for large, I/O bound problems that benefit from global data access by each compute node (cf. map/reduce). Our approach is complementary to map/reduce methods.

PVFS2 is designed to scale by aggregating network and I/O bandwidth across cluster resources. For many problems, performance *improves* with more nodes and larger data sets.

References

http://aws.amazon.com/ec2

http://pvfs.org

http://github.com/bwlewis/pvshm

http://www.r-project.org

http://cran.r-project.org/web/packages/bigmemory