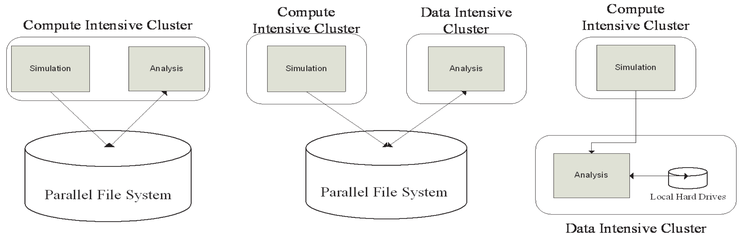
Compute and Data-Intensive Computing: Bridging the Gap

Grant Mackey, University of Central Florida Mentor: John Bent
Group: HPC-5 Email: gmackey@lanl.gov, johnbent@lanl.gov

Abstract

Scientific computing has generally composed of simulation workloads[5]. However, an emerging trend is data-intensive scientific computing; applications which are I/O dominated instead of being computationally bound[8,9]. Current HPC systems have different methods to cope with the I/O patterns of both application types[2,3,6,7]. However, current approaches incur issues of large data migration, severly impacting the time to completion of these scientific workloads. We propose a potential solution to this data migration overhead through the use of a distributed file system[1] which supports both compute and data-intensive I/O workload semantics.

Current Approaches



<u>Case A - Traditional HPC</u>) the following I/O operations are performed for HPC applications with both compute and analytics operations:

- **Compute-intensive cluster outputs its application results to conventional HPC storage**
- **Compute-intensive resource then uses HPC storage to read data for data-intensive analysis application**
- **Compute-intensive resource eventually outputs data to HPC storage.**

The last two steps are constantly repeated when running analysis applications.

<u>Case B - Adding a Data-Intensive Resource</u>)

- **Compute-intensive cluster outputs its application results to conventional HPC storage**
- **Output data is copied to a data-intensive resource and analysis is performed on the data-intensive resource**
- **Output of the data-intensive application is copied from data-intensive cluster to HPC storage**

Similarly to case A, in this approach, data is copied back and forth from HPC storage to data-intensive resource depending on the frequency of execution of an analysis operation.

Proposed Approach

Case C - Replace HPC Storage with a data-intensive resource to avoid the data migration among different resources. The I/O operations performed by using our approach are listed below:

- **Compute-intensive cluster outputs its application results to the data-intensive resource.**
- **Analysis is performed on the data-intensive resource, i.e. no data migration**

We present the theoretical analysis to compare our approach with the cases A and B. Assuming that:

1) **X = Time to write out simulation data with MPI/MPI-IO using a PFS**
2) **α = Copy/Data migration time from PFS to DI resource**
   - move simulation data to the data intensive resource for analysis
3) **β = Copy/Data migration time from DI resource to PFS**
   - move analyzed data off of the data intensive resource for storage
4) **Y_A = Data analysis time with conventional ADAT algorithm**
5) **Y_M = Data analysis time with MapReduce ADAT algorithm**
6) **Z = Time to write out simulation data using a DI resource**

***For our testing we utilize the USQCD suite. For our simulation we utilize the QIO/QMP package and for analysis we utilize ADAT[4].**

The total I/O times will be:

- **Case A: $X + \alpha + \beta + Y_A$**
- **Case B: $X + \alpha + \beta + Y_M$**
- **Case C: $Z + Y_M$**

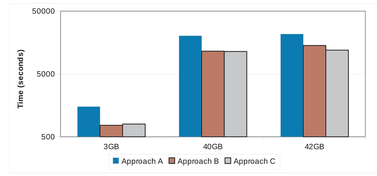
Comparing cases A and C, the analysis times will be different because we run two different analysis codes. While the algorithm is the same, they are run on two different systems, hence Y_A and Y_M.

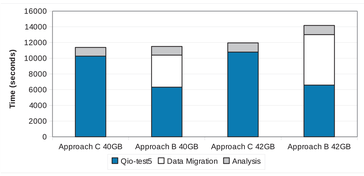
Our approach performs better when:

- **$X + \alpha + \beta + Y_A > Z + Y_M$, and**
- **$X + \alpha + \beta > Z$**

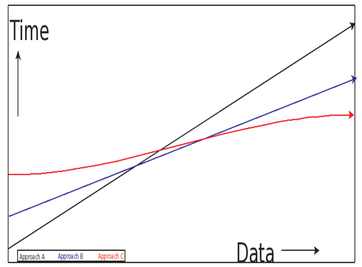
That is, our approach performs better than case A for instances where the analysis data cannot fit into the size of a compute intensive cluster's main memory, forcing it to make multiple large requests for data over a network to a parallel file sytem. Our approach performs better than case B for instances where α+β is sufficiently large.

Results





Initial results (above) show that the effects of data migration do impact the total execution time for a scientific workflow. As the size of simulation and analysis output grows, so does the amount of data to be shuffled between resources, and hence an increase in the total time to completion of a job for approaches A and B.



Theoretical results (left) show that there are appropriate conditions for all three approaches. The intersections of the three approaches illustrate at which points I/O becomes the bottle-neck for one approach or another. Further testing is needed to prove whether this formulaic approach holds true for real computing systems.

Conclusions

Our preliminary results are promising. We show that even with the overhead of a data-intensive file system which is not tuned for C.I. workloads, our approach takes less time and garners better overall performance. We show that for a comprehensive scientific workflow, replacing a parallel file system with a data intensive resource (removing data migration) results in 3.5x better job performance in preliminary testing. In the near future we will complete the scaled testing and verify the theoretical results.

Selected References

[1] Dhruba Borthaku. The Hadoop Distributed File System: Architecture and Design
[2] http://institutes.lanl.gov/isti/disc
[3] http://institutes.lanl.gov/isti/irhpit/projects/hdfspvfs.pdf
[4] http://usqcd.jlab.org/usqcd-software/
[5] http://www.lanl.gov/asc/docs/rr_open_science_final.pdf
[6] http://wiki.lustre.org/index.php/running_hadoop_with_lustre
[7] Rajagopal Ananthanarayanan, Karan Gupta, Prashant Pandey, Himabindu Pucha, Prasenjit Sarkar, Mansi Shah, and Renu Tewari. Cloud analytics: Do we really need to reinvent the storage stack? In HotCloud '09: Workshop on Hot Topics in Cloud Computing in conjunction with the 2009 USENIX Annual Technical Conference, 2009
[8] Tiankai Tu, Charles A. Rendleman, Patrick J. Miller, Federico Sacerdoti an Ron O. Dror, and David. E. Shaw. Accelerating parallel analysis of scientific simulation data via zazen. In FAST, pages 1–14. USENIX Association, 2010.
[9] YongChul Kwon, Dylan Nunley, Jeffrey P. Gardner, Magdalena Balazinska, Bill Howe, and Sarah Loebman. Scalable clustering algorithm for n-body simulations in a shared-nothing cluster. Technical report, University of Washington, Seattle, WA, 2009

Acknowledgements: The team managing the DISC cluster - Katherine Nystrom, HB Chen (HPC-5) and Ben McClelland (HPC-3) as well as Gary Grider (HPC-DO), Jim Ahrens (CCS-7)

LAUR: