Evaluating Information Dispersal Algorithms

ILLINOIS INSTITUTE OF TECHNOLOGY

Corentin Debains
Department of Computer Science
Illinois Institute of Technology
cdebains@hawk.iit.edu

Pedro Alvarez-Tabio
Department of Computer Science
Illinois Institute of Technology
palvare3@hawk.iit.edu

Ioan Raicu
Department of Computer Science
Illinois Institute of Technology
iraicu@cs.iit.edu

DataSys
Data-Intensive Distributed Systems Laboratory

Abstract

The explosion in data acquisition and storage has led to the emergence of data-intensive applications that are used to process enormous quantity of information using methods such as the MapReduce paradigm. Data-Intensive Distributed File Systems (DI-DFS) have been designed to support these kinds of applications. These large-scale storage systems require fault-tolerance mechanisms to handle failures, which are a norm rather than an exception when working at a scale that will shortly reach exascale. The old RAID definition has been replaced by replication in nowadays systems, but that implementation has many issues such as storage efficiency and durability. A new trend among large-scale systems is the implementation of information dispersal algorithms, called erasure codes. Those algorithms encode the data into multiple blocks among which only a portion is necessary to recover the original data. Several systems use this kind of algorithm and several libraries are available for system designers. The overhead introduced by the encoding and decoding can be a limiting factor for the integration at large-scale. At the same time, GPU computing has grown and now is used to perform non-graphical computations. In this project we compare two different approaches of erasure code computing, GPU and CPU. Therefore we compare two libraries, Gibraltar and Jerasure, which have proven to be efficient. We show a maximum of 5x performance increase that we can get from using GPU and also evaluate the performance impact of using the libraries at their limits with a growing number of storing nodes, e.g. the most reliable configuration possible. We aim to provide a good overview on erasure codes in order to help system designers integrate it in large-scale file-systems. Our work in erasure coding serves as the foundation for the next step: integrating an information dispersal algorithm that eventually outperforms current state-of-the-art approaches in DI-DFSs.

Goals

- Evaluate erasure coding libraries and their performance on Data-Intensive Distributed File Systems (DI-DFS)
  - Throughput, ideal parameters…
- Compare CPU vs GPGPU (General-Purpose GPU computing) in the particular case of erasure coding
  - How parallelizable can be made?
- Designing a full-fledged information dispersal system for a DI-DFS
  - This involves considering not only one node, but managing the metadata on n nodes
  - Important concepts: locality, replication for hybrid schemes

Performance evaluation

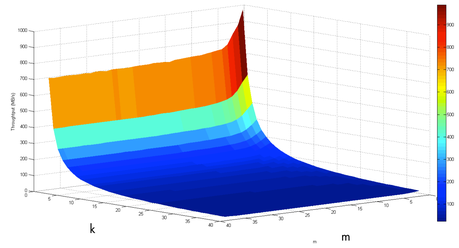


Figure 1a. Jerasure encoding

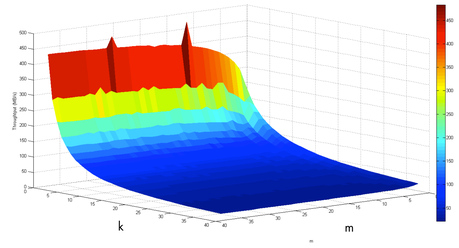


Figure 1b. Jerasure decoding

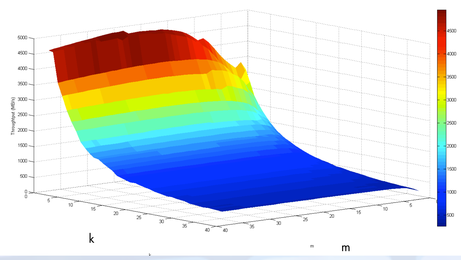


Figure 2a. Gibraltar encoding

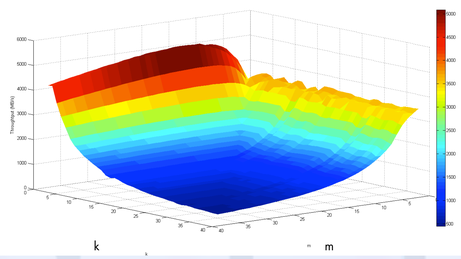


Figure 2b. Gibraltar decoding

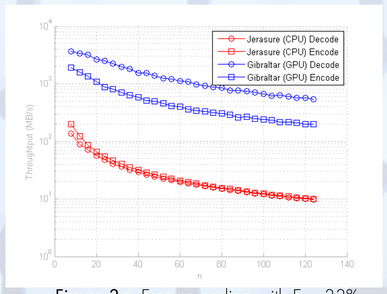


Figure 3a. Erasure coding with E = 33% and 1 MB buffer

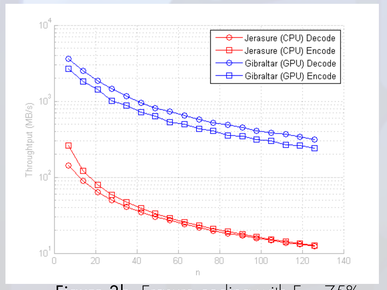


Figure 3b. Erasure coding with E = 75% and 1 MB buffer

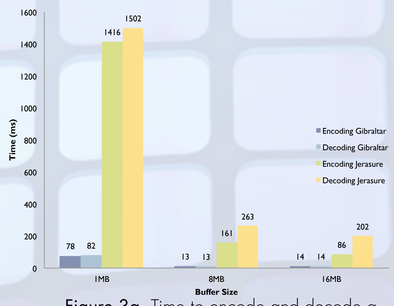


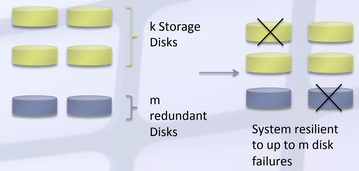
Figure 3a. Time to encode and decode a 64 MB file with an efficiency of 75%

Libraries used:
- Jerasure: leading erasure coding library, CPU computing, multi-algorithm encoding (RS, CRS…), fully customizable
- Gibraltar: CUDA-based library, less options (only RS coding), up to 5X performance!

Conclusions:
- GPU is the true enabler for erasure coding!
- Gibraltar has really promising performance
- Latencies while encoding and decoding a file are relatively low (20 ms at most when configured well), so erasure codes are definitely a good option for ensuring reliability and efficiency

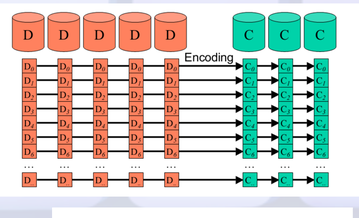
Erasure coding

You have k storage disks, and you want to tolerate m failures. Just encode m redundant disks.
- Generalization of RAID



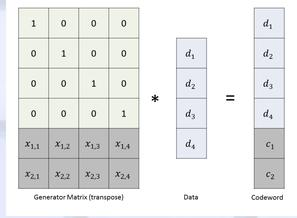
Reed-Solomon codes
- Each disk has a word size of w bits
- Focus on words, not entire disks → Object-based coding
- Word size conditions:

$$2^w \geq k + m$$



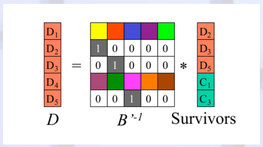
Encoding of data into codeword by Vandermonde matrices($x_{i,j}$ coefficients)
- Possible optimizations: Cauchy matrices, multiplications into XORs

For decoding, any k words are enough: best reliability

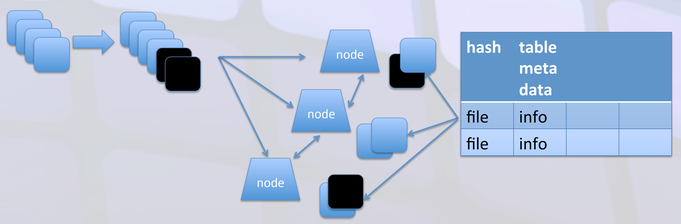


Important metrics: storage efficiency and failure tolerance

$$E_{storage} = \frac{k}{m+k} \qquad F_{tolerance} = \frac{m}{m+k}$$



Future work



- Towards a real Information Dispersal Algorithm: manage everything after erasure coding (data dispersal, metadata, response to failures)
- Adaptable container for CPU/GPU erasure coding (wrapper Jerasure and Gibraltar)
- Setting a framework for hybrid fault tolerance: replication (locality, performance) + information dispersal (efficiency, reliability)
- Including everything in FusionFS (DI-DFS currently under development at DataSys-IIT)

Related Material and Further Reading:

J.S. Plank, J. Luo, C. D. Schuman, L. Hu and Z. Wilcox-O'Hearn. "A Performance Evaluation and Examination of Open-Source Erasure Coding Libraries For Storage", in FAST-2009, San Francisco, CA, February, 2009.
J. S. Plank., S. Simmerman, C.D. Schuman. "Jerasure: A library in C/C++ facilitating era- sure coding for storage applications - Version 1.2" in Tech. Rep. CS-08-627, University of Tennessee, August 2008.
Matthew L. Curry, Anthony Skjellum, H. Lee Ward, Ron Brightwell. "Gibraltar: A Reed- Solomon coding library for storage applications on programmable graphics processors". Concurrency and Computation: Practice and Experience 23(18): 2477-2495 (2011)
I. Raicu, I. Foster, P. Beckman. "Making a Case for Distributed File Systems at Exascale," in Proc. of ACM Workshop on Large-scale System and Application Performance (LSAP), 2011.