DOI: 10.14529/jsfi160304

Analyzing Data Properties using Statistical Sampling – Illustrated on Scientific File Formats

Julian M. Kunkel[1]

Understanding the characteristics of data stored in data centers helps computer scientists in identifying the most suitable storage infrastructure to deal with these workloads. For example, knowing the relevance of file formats allows optimizing the relevant formats. It also helps in a procurement to define benchmarks that cover these formats.

Existing studies that investigate performance improvements and techniques for data reduction such as deduplication and compression operate on a subset of data. Some of those studies claim the selected data is representative and scale their result to the scale of the data center. One hurdle of running novel schemes on the complete data is the vast amount of data stored and, thus, the resources required to analyze the complete data set. Even if this would be feasible, the costs for running many of those experiments must be justified.

This paper investigates stochastic sampling methods to compute and analyze quantities of interest on file numbers, but, also, on the occupied storage space. It will be demonstrated that on our production system, scanning 1% of files and data volume is sufficient to deduct conclusions. This speeds up the analysis process and reduces costs of such studies significantly.

Keywords: Scientific Data, Compression, Analyzing Data Properties.

Introduction

Understanding the characteristics of data stored in the data center helps computer scientists to optimize the storage. The quantities of interest could cover proportions, i.e. the percentage of files with a certain property or means of certain metrics, such as achievable read/write performance, compression speed and ratio. For example, knowing the relevance of file formats may shift the effort towards the most represented formats. When 80% of the capacity is utilized by NetCDF4 files, performance analysis and optimization should target this file format first. Understanding the achievable compression ratio of available compression schemes helps in choosing not only the best one for user-specific compression, but also for file system level compression.

In the literature, one can be found studies that investigate compression ratio, de-duplication factor or improve performance of scientific middleware. Due to the long running time to apply any improvement on large amounts of data, many studies assume the benefit measured on a small data sample can be transferred to the scale on the data center. However, usually in these studies nobody pay attention if the data set is actually representative. In other words, they do not take into account the fraction of the workload that can actually benefit from the advancement. In statistics, the big field of sampling theory addresses this issue. Due to the law of large numbers, there are methods to draw instances appropriately and deduce properties from the sample set to the population with high confidence. However, this process is non-trivial and a research discipline in statistics by itself [3].

This paper investigates statistical sampling to estimate file properties on the scale of data centers using small data sets and statistical simulation. The computation time used for the complete project was 517 core days. With 24 cores per node, a complete system scan of DKRZ's

[1] Deutsches Klimarechenzentrum, Bundesstraße 45a, 20146 Hamburg, Germany

system would have needed about 475 node days which would have cost at least about 4000[2] – while not revealing additional insight. Instead with 1% of scanned files or capacity, similar results are achievable , that means with 1% of scanned files we have confidence that we estimate sufficiently accurate the characteristics of the full system. If scanning all files has no impact on the conclusions we draw (e.g., we can save 51% of storage with compression A), then why should we have to scan all files which is a snap-shot of the system life, anyway? . Note that a significantly extended version of this paper containing results for compression will appear in [5].

1. State of the art

Existing research that analyzes properties of scientific data can be classified into performance analysis, compression ratio and data deduplication. The effort that investigates and optimizes performance usually picks a certain workload to demonstrate that the new approach is superior than existing strategies. A few studies analytically analyze typical patterns and optimize for a large range of access patterns. An example is the study in [8], which analyzes the access pattern for several workloads and discusses general implications. The research on optimization techniques, as far as known to the author, do not check how many people actually benefit from these optimizations and the implications on the system level.

In the field of compression, many studies have been conducted on pre-selected workloads, for example, see [1, 4, 6]. Some of those studies are used to estimate the benefit of the compression on the data center level, for example, Hübbe et al. investigate the cost-benefit for long-term archival. Similarly in [4], the compression ratio is investigated. However, in this case the selected data is a particular volume from a small system.

Modern file systems such as BTRFS and ZFS offer compression on system-side [7]. It is also considered to embed compression into storage devices such as SSDs [10] and evaluate it for OLTP workloads. In [2], Jin et.al investigate the benefit for compressing and de-duplicating data for virtual machines. They created a diverse pool of 52 virtual images and analyzed the impact.

As far as known to the author, statistical sampling techniques have not been used to investigate file characteristics for data centers.

2. Sampling of Test Data

To assess and demonstrate the impact of statistical sampling, firstly, a subset of data of DKRZ's supercomputer Mistral is scanned and relevant data properties about data compression and scientific file types are extracted. Our global Lustre file system hosts about 320 million files and 12 Petabytes of space is occupied; only a subset is scanned: 380k (0.12%) accounting for an (aggregated size) of 53.1 TiB of data (0.44%). To prevent data loss and ensure data privacy, the scanning process is performed using a regular user account and, thus, it cannot access all files. There are still 58 million files and 160 out of 270 project directories are accessible.

The scanning process used as a baseline in the paper works by running find to scan all accessible files, then select 10k files from each project randomly in a candidate list for the scans. Then, the list is permuted and partitioned into different threads. Multiple worker threads are started and each thread processes its file list sequentially running 1) the CDO [9] command

[2] Considering the TCO of purchasing (35M € and operating the system into account (more than 10M € for 5 years).

to identify the scientific file type, 2) the `file` command to guess the file types, and a list of compressors (LZMA, GZIP, BZIP2, ZIP). The `time` command is used to capture runtime information of each step and reported user time is used to assess performance. When 300k files are scanned, the threads are terminated and the results are collected.

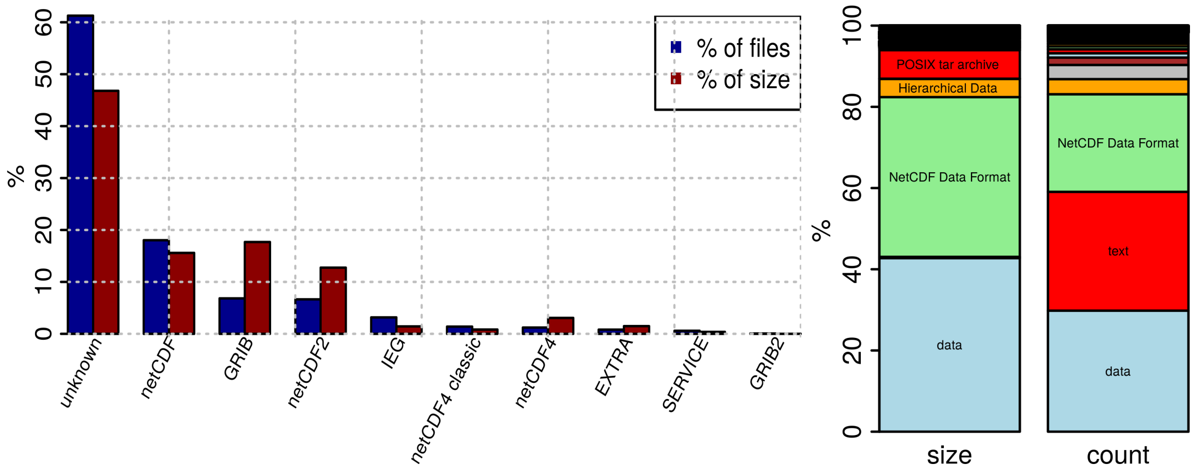
The strategy increases the likelihood that a representative sample of files is chosen from accessible projects. It is expected that projects differ in their file characteristics. The goal of the strategy is not to gain a completely representative sample, since this is to be developed within this paper. The limitations of this sampling strategy to investigate properties based on the occupied file size will be shown later.

Albeit the analysis described in the following sections are achieved with the suboptimal sampling, they correctly simulate the behavior of a system and show that the method delivers the correct results. But it means, that the obtained characteristics computed do not predict DKRZ's full data set correctly. We are currently applying the correct sampling technique on the full data set to identify the true characteristics for DKRZ.

3. Difference in Means Computed by File Count and File Size

This section gives an example to understand that the results vary depending on whether metrics are computed either on file count, i.e. each file is weighted identically, or by weighting each file with its occupied size.

The usage of scientific file formats is shown in Figure 1. The figure shows the relative relevance in terms of occupied space and a number of files of each file format. About 60% of the number of files and 47% of aggregated file size is non-scientific and cannot be resolved with the CDO tool. The dominant scientific formats are NetCDF3, GRIB1 and NetCDF2. The `file` command cannot identify and distinguish scientific formats as reliable as CDO, but can shed light over the distribution of the 60%. Looking at its output, the 60% of capacity seems to be dominated by TAR (7%) and GZIP compressed files (5%) – it classifies 43% of capacity as "data" and 40% as NetCDF (no version information provided). Looking at the proportions in terms of file count, roughly 30% are classified as data, 30% as text (e.g., code), 24% as NetCDF files, 4% as HDF5, and 3.5% as images. Other file types are negligible.



(a) CDO types (b) File types

Figure 1. Relative usage of file formats determined using CDO and `file`

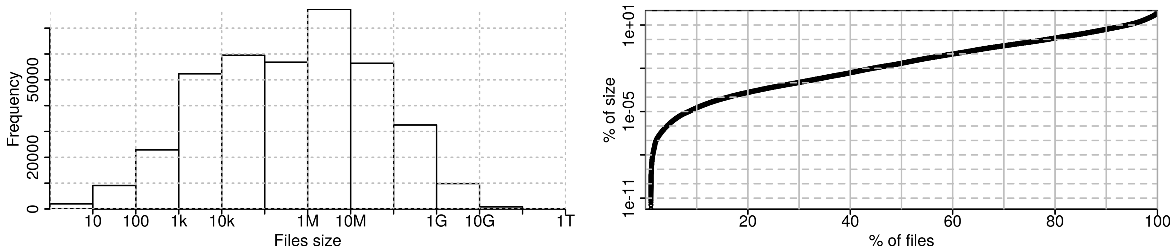
The difference between the proportion computed by the file count and by file size stems from the highly skewed distribution of file sizes and the different mean size across different file formats. Fig. 2 show the distribution of file sizes. Fig 2a) shows a histogram with logarithmic file sizes. In Fig. 2b) the relation between the file size and the file count is illustrated; to construct the figure, files have been sorted by size in ascending order and then the cumulative sum is computed. While the histogram suggests similarities between size distribution and a normal distribution, this is due to the logarithmic x-axis. In the cumulative view, it can be seen that aggregated 20% of files consume one millionth of storage space and 90% still consume less than 10% space. If a study takes small files as representatives, those fail to represent the storage capacity. Similar large files fail to represent the typical (small) file that must be handled by the storage.

4. Stochastic Sampling of Data

The way the quantities of interest are computed are either by file count, i.e. we predict properties of the population based on individual files, or by weighting the individual files with their size. From the perspective of statistics, we analyze variables for quantities that are continuous values or proportions, i.e. the fraction of samples for which a certain property holds. To select the number of observations that allows inference about the population, statistics knows methods for determining a sample size. More information about this topic is provided in the full paper [5].

Sampling method to compute by file count. When computing the proportion or the mean of a variable for files, a strategy is to enumerate all files on the storage system and then create a simple random sample, i.e. choose a number of files for which the property is computed.

Sampling method to compute by file size. Estimating values and weighting them based on the file size requires to enumerate all files and determine their size, then pick a random sample from the file list based on the probability defined by filesize/totalsize. Draws from the list must be done with replacement, i.e. we never remove any picked file. Once all chosen files are determined, the quantities of interest are computed once for each unique file. Then, each time we have chosen a file, we add our quantity of interest without weighting the file size, for example, the arithmetic mean can be computed just across all samples. Thus, large files are more likely to be picked but each time their property is accounted identically as for small files.



(a) Histogram (b) Cumulative file sizes (y-axis in log scale)

Figure 2. Distribution of file sizes

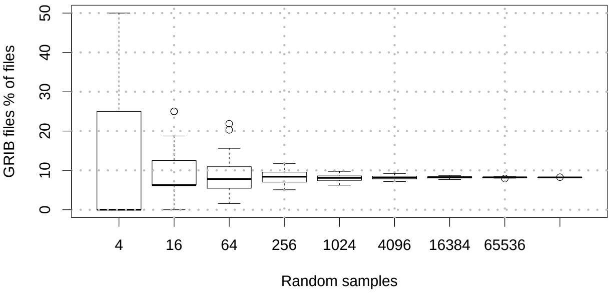
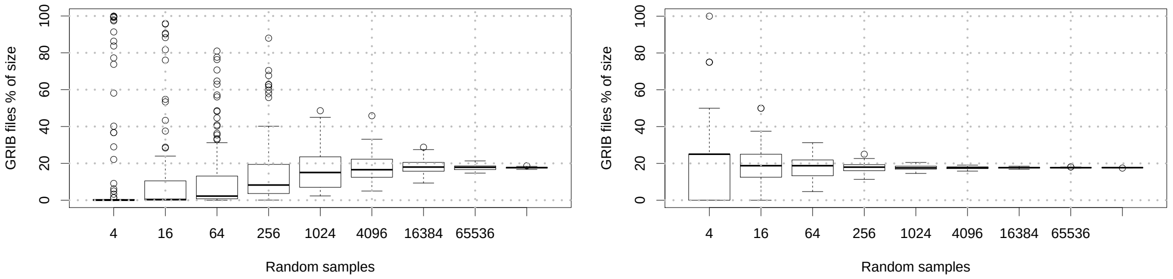


Figure 3. Simulation of sampling by file count to compute compr.% by file count



(a) By file count (this is suboptimal!). (b) Correct sampling proportional to file size.

Figure 4. Simulation of sampling to compute proportions of types by size

Robustness. To illustrate the stability of the approach, a simulation has been done by drawing a variable number of samples from the data. The simulation is repeated 100 times and a boxplot is rendered with the deviations. Naturally, the repeats of a robust method should have little variance and converge towards the correct mean value. The result for the proportion of GRIB files are given as an example, but the results for all variables behave similar. In Figure 3, it can be clearly seen that the error becomes smaller.

The sampling strategy to compute quantities on file size is shown in Figure 4b). Similarly, to the correct method for sampling by file count it converges quickly. However, if we simply use a file scanner to compute the metrics on size, but it would choose files randomly without considering file sizes, we would achieve highly unstable results (Figure 4a). Indeed, the error margin with even one fifth of all files (64k) is comparable to the correct sampling strategy with only 1024 samples. Thus, it is vital to apply the right sampling method. Therefore, the initial approach used to gather the test data as described in Section 3 is suboptimal.

Summary & Conclusions

In this paper, sampling techniques from statistics are applied to estimate data properties. These techniques are demonstrated to be useful approximate the proportions of scientific file types. It has been demonstrated that a random file scanner is not efficient to estimate quantities that are computed on file size. Instead, sampling with replacement and a probability equal to the proportion of file size leads to stable results. The tools which use such techniques can estimate properties of data robust without the need to analyze the huge data volumes of data centers. We will be working on such tools to evaluate the benefit of optimization strategies. More results are found in the full paper [5].

References

1. Nathanel Hübbe and Julian Kunkel. Reducing the HPC-Datastorage Footprint with MAFISC – Multidimensional Adaptive Filtering Improved Scientific data Compression. *Computer Science - Research and Development*, pages 231–239, 05 2013. DOI: 10.1007/s00450-012-0222-4.

2. Keren Jin and Ethan L Miller. The Effectiveness of Deduplication on Virtual Machine Disk Images. In *Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference*, page 7. ACM, 2009.

3. JWKJW Kotrlik and CCHCC Higgins. Organizational Research: Determining Appropriate Sample Size in Survey Research Appropriate Sample Size in Survey Research. *Information technology, learning, and performance journal*, 19(1):43, 2001.

4. Michael Kuhn, Konstantinos Chasapis, Manuel Dolz, and Thomas Ludwig. Compression By Default – Reducing Total Cost of Ownership of Storage Systems, 06 2014.

5. Julian M. Kunkel. Analyzing Data Properties using Statistical Sampling Methods – Illustrated on Scientific File Formats and Compression Features. In *High Performance Computing – ISC HPC 2016 International Workshops, Revised Selected Papers (to appear)*, volume 9945 of *Lecture Notes in Computer Science*. 2016.

6. Sriram Lakshminarasimhan, Neil Shah, Stephane Ethier, Seung-Hoe Ku, Choong-Seock Chang, Scott Klasky, Rob Latham, Rob Ross, and Nagiza F Samatova. ISABELA for Effective in Situ Compression of Scientific Data. *Concurrency and Computation: Practice and Experience*, 25(4):524–540, 2013. DOI: 10.1002/cpe.2887.

7. Solomon Desalegn Legesse. Performance Evaluation of File Systems Compression Features. Master's thesis, University of Oslo, 2014.

8. Jay Lofstead, Milo Polte, Garth Gibson, Scott Klasky, Karsten Schwan, Ron Oldfield, Matthew Wolf, and Qing Liu. Six Degrees of Scientific Data: Reading Patterns for Extreme Scale Science IO. In *Proceedings of the 20th international symposium on High performance distributed computing*, pages 49–60. ACM, 2011. DOI: 10.1145/1996130.1996139.

9. Uwe Schulzweida, Luis Kornblueh, and Ralf Quast. CDO User's guide: Climate Data Operators Version 1.6. 1, 2006.

10. Aviad Zuck, Sivan Toledo, Dmitry Sotnikov, and Danny Harnik. Compression and SSDs: Where and How? In *2nd Workshop on Interactions of NVM/Flash with Operating Systems and Workloads (INFLOW 14)*, Broomfield, CO, October 2014. USENIX Association.