Computing Just What You Need: Online Data Analysis and Reduction at Extreme Scales

Ian Foster[1,2], Mark Ainsworth[3], Bryce Allen[2], Julie Bessac[1], Franck Cappello[1], Jong Youl Choi[4], Emil Constantinescu[1], Phillip Davis[6], Sheng Di[1], Wendy Di[1], Hanqi Guo[1], Scott Klasky[3], Kerstin Kleese Van Dam[5], Tahsin Kurc[7], Abid Malik[5], Kshitij Mehta[4], Klaus Mueller[7], Todd Munson[1,2], George Ostouchov[4], Manish Parashar[6], Tom Peterka[1], Line Pouchard[5], Dingwen Tao[1], Ozan Tugluk[3], Stefan Wild[1], Matthew Wolf[3], Justin Wozniak[1], Wei Xu[5], and Shinjae Yoo[5]

[1] Argonne National Laboratory, Lemont, Illinois, USA
[2] University of Chicago, Chicago, Illinois, USA
[3] Brown University, Providence, Rhode Island, USA
[4] Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA
[5] Brookhaven National Laboratory, Brookhaven, New York, USA
[6] Rutgers University, New Brunswick, New Jersey, USA
[7] Stony Brook University, Stony Brook, New York, USA

Abstract. A growing disparity between supercomputer computation speeds and I/O rates makes it increasingly infeasible for applications to save all results for offline analysis. Instead, applications must analyze and reduce data online so as to output only those results needed to answer target scientific question(s). This change in focus complicates application and experiment design and introduces algorithmic, implementation, and programming model challenges that are unfamiliar to many scientists and that have major implications for the design of various elements of supercomputer systems. We review these challenges and describe methods and tools that we are developing to enable experimental exploration of algorithmic, software, and system design alternatives.

1 Introduction

Technology trends are creating a crisis in high performance computing. Computer speeds are increasing much faster than are storage technology capacities and I/O rates. For example, the Mira supercomputer installed at Argonne National Laboratory in 2012 has a peak compute rate of 10 petaflop/s (10^{16} op/s) and disk write rate of 500 GB/s (5×10^{11} bytes/s). By 2024, computers are projected to compute at 10^{18} ops/sec but write to disk only at 10^{12} bytes/sec: a compute-to-output ratio 50 times worse. Figure 1 provides another perspective on this trend. We can no longer output every piece of information that we might ever possibly *want*. Instead, we need to output just the information that we *need* to answer some question(s). This new goal requires new thinking about the design and implementation of both applications and system software.

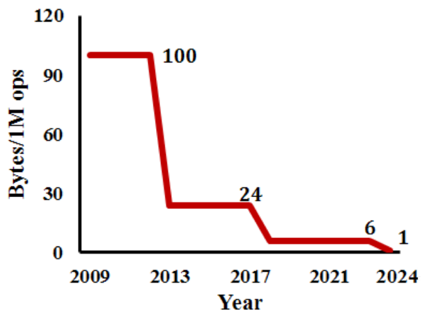

Fig. 1: Total filesystem throughput of leadership class facilities vs. total floating point operations per second [28,35,47]. I/O throughput scales more slowly than computational speed.

Increasing use of supercomputers for near-real-time decision making is another factor motivating new thinking about application and system software design. For example, both experimental fusion energy experiments and next-generation light sources are moving to a new frontier where data must be processed rapidly to enable near-real-time decisions. It is no longer feasible to input experimental data, perform some computation (e.g., simulation of the experiment's future trajectory), and then output results for later analysis. Data must be processed immediately to detect significant events (e.g., vortices in a fusion experiment, features in microtomography) in order to permit rapid feedback to the experiment.

In both purely computational and coupled experimental–computational studies, these growing disparities between computational speeds and I/O rates demand new application structures that link previously disjoint activities: experiment, simulation, data analysis, data reduction. Yet while many algorithms and tools exist to treat separate pieces of such problems, these capabilities are often inoperable or inaccessible to the research scientist. Scientists need new tools for coupling components and new methods for co-optimizing the resulting workflows. These tasks introduce algorithmic, implementation, and programming model challenges that are unfamiliar to many scientists and that have major implications for the design of various elements of high performance systems.

The Codesign center for Online Data Analysis and Reduction (CODAR) engages scientists at three national laboratories and four partner universities, to address these challenges. Working closely with applications teams, CODAR is undertaking a codesign process that targets both common data analysis and reduction methods (e.g., feature and outlier detection, and compression) and methods specific to particular data types and domains (e.g., particle and structured finite-element methods). Our goal is to understand and guide tradeoffs in the development of computer systems, applications, and software frameworks, given constraints relating to application development costs and fidelity, performance portability, scalability, and power efficiency, and to answer these questions:

Q1: What are the best data analysis and reduction algorithms for different application classes, in terms of speed, accuracy, and resource needs? How can we implement those algorithms for scalability and performance portability?

Q2: What are the tradeoffs in data analysis accuracy, resource needs, and overall performance between online reduction and offline analysis vs. online analysis? How do these tradeoffs vary with hardware and software choices?

Q3: How do we effectively orchestrate online data analysis and reduction to reduce associated overheads? How can hardware and software help?

2 Example applications

We use three examples to motivate the need for online data analysis and reduction.

2.1 Climate science

Climate scientists want to run large ensembles of high-fidelity 1km$\times$1km simulations on exascale systems, with each instance simulating 15 years of climate in 24 hours of computing time. They estimate that outputting the full model state for each ensemble member once per simulated day would generate 260 TB every 16 seconds across the ensemble, approximately 16$\times$ what can be written to the parallel file system at the expected peak output rate of 1 TB/sec. (Currently, climate models achieve much lower I/O rates, due to the relatively small size of model grids.) Furthermore, even following data reduction to 1 TB/sec, such model runs would output an astonishing 85 PB per day, posing major storage and offline data analysis challenges.

While 85 PB is a lot of data to output in a day's computing, this quantity represents just a small subset of the total data to be produced by the ensemble. Outputting state just once per simulated day represents a highly lossy reduction, given that the climate model time step may be just 100 simulated seconds, And indeed dome analyses may require access to the full state at higher frequency. For example, feature detection (e.g., tracking cyclones, detecting areas of extreme heat) may require access to model state once per simulated five minutes, a rate 24$\times$12=288 greater. Clearly, climate models need new online data analysis and reduction methods that can both preserve more information than once-per-day snapshots and produce considerably less data.

2.2 Fusion science

Fusion scientists are developing a high-fidelity whole device model for magnetically confined fusion plasmas, for use in planning experiments on the ITER facility [1] and simulating future experimental fusion devices [7]. The X-point included Gyrokinetics Code (XGC) [27], one potential component of a whole device model, models the plasma edge. A single XGC simulation can produce hundreds of petabytes of data describing particle positions and the state of the field within which the particles move.

We use this example to illustrate the need for application-aware data reduction methods. To reduce this data to manageable sizes, ultimately allowing 100 PB to be reduced to 100 TB, a 1000:1 reduction, fusion scientists and CODAR participants collaborated to devise a multistep data reduction process. The first step was to simply decrease output frequency. However, this approach cannot be taken beyond physically relevant time scales; important information would be lost by decreasing the frequency further. The second step was to use application knowledge to further reduce the data without losing essential information. The XGC particles are assumed to follow a Maxwellian distribution.

Therefore, we fitted a distribution to the data and saved the parameters for the distribution and the particles falling outside that distribution (the “outliers”). For the field data, adaptive data reduction methods were used to preserve features (see Figure 2). Finally, generic compression methods were applied to achieve further data reduction. The reduced data was then output and used for offline data analysis.

2.3 Materials science

Materials scientists regularly run billion-atom atomistic simulations with femtosecond timesteps on leadership-class machines [38, 44]. In order to understand phenomena such as the structural properties of lignin-based macromolecules, information essential for improving biofuel production, measurable vibrational responses that arise at the tens of femtoseconds must be studied, requiring per-timestep data access. Folding and bonding properties, however, arise only on the scale of seconds. Saving the full state to simultaneously study both quantities would generate exabytes on exascale computers. Intelligent, statistically valid spatial and temporal data analyses and/or reductions that can be applied online are needed to achieve accurate scientific characterizations without the need to save full-resolution data.

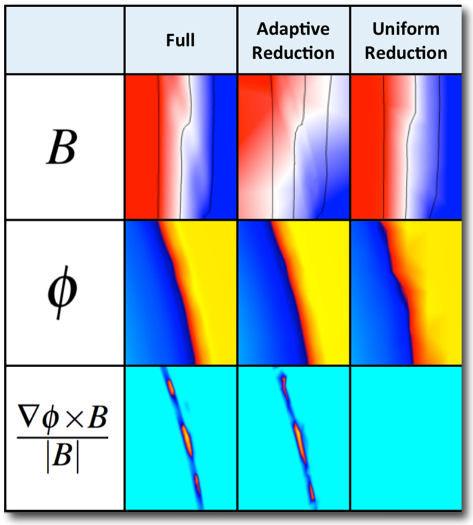

Fig. 2: XGC fusion simulation results near the plasma edge illustrates the need for fidelity preserving data reduction. The full data for the magnetic field $||B||$ and the scalar potential ϕ both show close approximation to the full solution. However, in the case of the derived fluid velocity $\frac{\nabla\phi\times B}{||B||}$, the adaptive method retains the four major features from the full data; the uniform method does not.

3 A high-performance codesign architecture

Some science teams have already developed application-specific online data analysis and/or reduction methods on petascale systems, methods they now need to scale for exascale. Others face the prospect of having to integrate such methods from scratch as part of their preparation for exascale. In both cases, we want to make it easy for them to integrate a variety of scalable online data analysis and reduction methods into their existing infrastructure, so that they can easily experiment with co-design alternatives and achieve performance portability.

3.1 The need for modular implementations

A first key to achieving this goal, we believe, is to modularize implementations so that analysis and reduction methods, resource allocations, and coupling methods can be varied with little or no changes to an application. In this way we facilitate experimentation with design alternatives and investigation of codesign and performance portability questions.

The key to modular integration of applications with online data analysis and reduction methods is access to both the application data of interest and metadata that describe those data's structure. Once this access is enabled, it becomes straightforward to access and exchange the data to be analyzed and/or reduced. Our team has much experience in instrumenting applications to provide and use such information, particularly in the context of the Adaptable IO System (ADIOS) [19, 51] and the Swift [50] system, but also in earlier work [14, 52]. In many cases, this instrumentation involves adding simple procedure calls, for example via the ADIOS application program interface (API) [31], to the application to indicate the data structures in question. A runtime system can then extract the specified data and pass it to specified data analysis and reduction services. Only the runtime implementation, not the application, needs to be modified to explore alternative implementation strategies, such as processing on the same or different nodes, using NVRAM, varying clock speeds for power efficiency, or varying the number of data analysis nodes.

3.2 CODAR system components

These considerations lead us to identify three major classes of CODAR co-designed technologies (see Figure 3 for an illustration of how they fit together).

The **CODAR Data API** provides the means by which applications specify the data to be analyzed and/or reduced and its structure. We leverage the ADIOS API [26], which has been integrated into more than thirty science applications [19, 33, 45, 51]. One co-design question will be how to extend this API so that applications can convey actionable information for exascale optimizations relating to performance and power efficiency.

CODAR Data Services provide scalable implementations of data analysis and reduction methods, plus ancillary monitoring methods, all packaged to permit their use by any application. The data reduction methods will provide effective reduction of the simulation outputs, both the application state and the results of online data analyses applied to that state, while retaining simulation fidelity. Data monitoring is needed to verify that a particular data reduction method is retaining the necessary information and to provide feedback when the data reduction is either too aggressive or not aggressive enough (see Figure 2). These services will include a mix of those developed by us and those imported from other sources. Realistically, we will produce only a modest number of such implementations ourselves, but our methods and co-design knowledge will be broadly applicable. Anyone will be able to add generic or application-specific

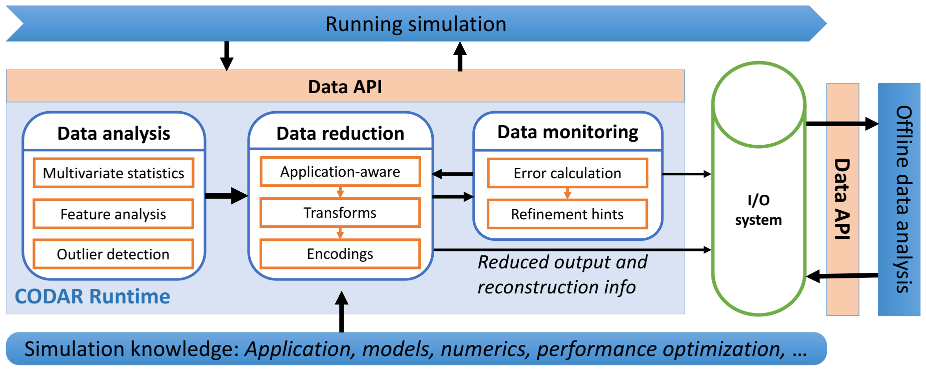

Fig. 3: Prototypical data analysis and reduction pipeline, showing how a simulation communicates to our services through an API that conveys data and their structure.

data services. An important co-design question here concerns the methods and support required for efficient execution of a broad range of such services.

The **CODAR Runtime** provides methods for the deployment, configuration, execution, and computational monitoring of applications and associated data analysis and reduction pipelines on exascale platforms. Given a specified set of data analysis, reduction, and monitoring services, it will enable their efficient composition and configuration; their deployment to appropriate nodes and cores; efficient communication among them; computational monitoring of both individual services and the complete computation; and adaptation of service configurations and parameters.

These three sets of co-designed technologies, each to be delivered as open-source software, will allow application teams, working with or without our application catalysts, to instantiate versions of the Figure 3 pipeline to address their specific science goals. Lessons learned from experiments with diverse applications, methods, and platforms will in turn feed back to ECP application projects, software projects, vendors, and other stakeholders.

4 The CODAR runtime

The CODAR Runtime provides methods for controlling the placement and configuration of CODAR Services for purposes of co-design exploration and performance optimization. The initial focus is on simple manual configuration of service delivery choices; in later stages of the project, we will also provide for automated configuration, once co-design strategies are better understood. Figure 4 shows the initial set of components.

The **Cheetah** experiment management framework defines a set of conventions and re-usable scripts for conducting parameter sweep experiments on different science applications. Such experiments are intended to be run on supercomputers, particularly on existing machines, but may also be run on local workstations for debugging. An ‘application’ may be a single science code or, more typically, one or more science codes plus a set of online analysis and reduction codes that are coupled with the science codes and each other. The goal of such parameter sweep experiments is to determine the best set of parameters to use to run the application as efficiently as possible on different target machines. This 'best' set of parameters usually varies over different machines.

The **Savannah** in situ runtime:

– Provides a tested deployment framework for any application (or software technology) project to utilize online data analysis and reduction.
– Provide the infrastructure needed to create a testing framework (Cheetah) to evaluate reduction and analysis functions for performance on a variety of levels (application and platform)
– Provide a reference approach for teams that have specialized needs that exceed the infrastructure design constraints.

Savannah is not intended to be the only possible way of deploying CODAR-developed or vetted analytics and reduction functions; multiple cooperating ecosystems are needed to make the total system thrive. However, Savannah offers a convenient and straight-forward approach, making it easier for applications to focus on the science, rather than the details of advanced scheduler settings, rdma network transfers, and other technical details that tend to interfere with the deployment of online techniques.

Finally, the **Chimbuko** performance data capture suite captures, analyzes and visualizes performance metrics for complex scientific workflows and relates these metrics to the context of their execution on extreme-scale machines to enable empirical performance studies. Because capturing performance metrics can quickly escalate in volume and provenance can be highly verbose, Chimbuko interfaces with (lossy) data compression modules specialized for high-velocity performance data.

To quantify co-design tradeoffs involved in online data analysis and reduction for a particular application, an ensemble of executions would be run using

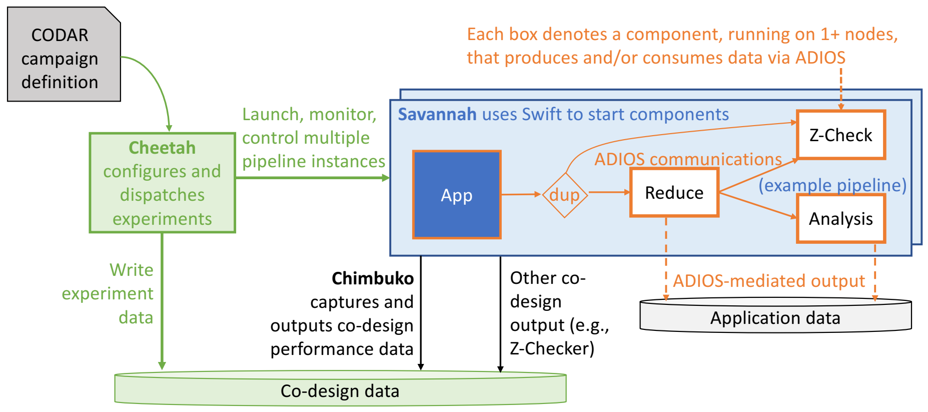
CODAR campaign definition
Each box denotes a component, running on 1+ nodes, that produces and/or consumes data via ADIOS
Cheetah configures and dispatches experiments
Launch, monitor, control multiple pipeline instances
Savannah uses Swift to start components
Z-Check
ADIOS communications
App
dup
Reduce
(example pipeline)
Analysis
Write experiment data
Chimbuko captures and outputs co-design performance data
Other co-design output (e.g., Z-Checker)
ADIOS-mediated output
Application data
Co-design data

Fig. 4: The CODAR codesign system, showing in particular the Cheetah experiment management component and the Savannah runtime.

Cheetah and Savannah, each involving an application X plus an analysis A and a reduction R (e.g., from Z-checker) with different specifications of the information that needs to be saved when (e.g., different data reduction mechanisms and parameters) and what work is to be placed where (e.g., different numbers of nodes allocated to X, A, and R; X, A, and R allocated to the same or different nodes; and different mechanisms used to transfer data between components). Chimbuko would capture the performance information for each member of the ensemble and enable analysis across the ensemble to answer co-design questions.

5 CODAR data services

We shows in Figure 3 a simplified view of online data analysis in which a process consumes simulation data and produces extracted information that is communicated back to the simulation and/or sent to a data reduction service for further processing prior to storage on the parallel file system. More generally, data analysis methods may extract information from several states—for example, a sliding window of time—and use results from previous data analysis methods. Our Data Services are intended to provide the abstraction and connection to the Data API to implement data analysis methods required by applications and to allow the composition of these methods via the CODAR Runtime to produce data analysis workflows.

5.1 Analysis services

Our initial catalog of data analysis methods will concentrate on three broad areas: (1) multidimensional statistical and image analysis, (2) topological analysis, and (3) outlier detection and extraction. We will develop this set based on ECP application requirements and their relevance to important co-design questions, such as the following. When should a data analysis be performed online versus offline? How frequently can data analyses be performed online, given a specified computational budget? How can data analyses make use of increased CPU on-node concurrency? When do we use burst buffers to stage and extend memory for online data analysis? How do we take advantage of deep memory hierarchies for tracking changes over time?

Multidimensional statistical and image analysis Application scientists frequently find it useful to extract multidimensional statistics and geometrical characteristics from simulations, since these analyses reflect properties on a larger scale than do pointwise and time-instant measurements and carry information about structures, aggregated quantities, and statistical measurements. We will integrate implementations of powerful techniques such as template matching and segmentation methods to extract structures from multidimensional data. One example is Markov random field methods that employ clique analysis to identify structural arrangements of data. Such methods have been parallelized over a large number of cores through graph partition methods [36].

Such methods can be applied, for example, to track the evolution of structures in time in materials science. Analyzing the formation and progression of cracks, connectivity lines, pores, and channels in time-lapse data can require extracting global features. Advanced feature extraction applied to experimental data is currently an active area, with considerable work [37, 48] relying on template matching, parallel Markov field methods, and statistical priors.

We will also build on our stochastic flow map [17], which provides understanding of uncertain transport behavior. This map has been successfully applied to climate [17, 40] and weather [16] applications. We will further develop our data analysis methods to model multivariate and multiscale features in statistical ensembles using the concepts of specific mutual information between variables [6] and information flows based on association rules [32]. These methods all have a wide range of applications including climate and combustion.

As an example, climate model ensembles produce a distribution of velocities, instead of a single velocity at each grid point. These distributions allow climate scientists to quantify the uncertainty in convergent and divergent transport behaviors and in derived features such as eddies, flow segmentation, and large-scale teleconnections. Tracking these features via stochastic flow maps enables scientists to understand their evolution and advance their scientific mission.

Topological analysis Extracting stable topological features, such as components (e.g., halos in cosmological simulations), loops, and voids is vital to many applications. We will integrate persistent homology methods [12,13] to identify stable features in the data. Their distributions will be summarized in persistence diagrams, which are point sets that quantify how much the data needs to change in order to eliminate any given feature [8]. These diagrams are equipped with natural metrics [25] (e.g., bottleneck and Wasserstein distances) and are stable to perturbations of the data with respect to those metrics. We will then measure the rate of change of topology as the simulation progresses and use this rate to measure topological deviation.

As an example, an important analysis technique in cosmology involves building catalogs of dense clusters of matter, called halos, profiling the distribution of their masses, and studying their evolution over time. These catalogs allow scientists to compare simulations to each other and to observations, in order to help identify unknown cosmological parameters. Improved data analysis methods that incorporate richer topological structures present in the simulations and use more discriminative comparison methods are essential for cosmological codes to extract maximum scientific insight.

Outlier detection and extraction Outliers and rare events are the needles that application scientists frequently seek in the massive haystack of exascale data. We will develop semi-supervised machine learning techniques that incorporate existing prior knowledge (such as a Bayes classifier) within an unsupervised learning algorithm to select the most relevant targets for later inspection and addition to a corpus of information. We will integrate the iForest [22] unsupervised machine learning algorithm to project data into a subspace where outliers deviate sharply from the remaining data, and we will apply kernel-based signatures to detect outliers [20–22]. This combination is particularly effective in the case of complex data with extremely high dimensionality [21, 22].

5.2 Reduction services

As illustrated in §2, the communication, analysis and storage of data from exascale simulations will only be possible through aggressive data reduction capable of shrinking datasets by one or more orders of magnitude. Such data reduction level is not feasible with lossless data reduction (e.g., lossless compression) that only typically achieve reduction factors of 2 (initial size/reduced size) on scientific data. Only lossy data reduction has the potential to reach reduction factors of orders of magnitude.

As shown in Figure 3, online data reduction services consume both simulation outputs and the results from online data analyses and prepare the data to be written to the parallel file system. A crude but commonly used data reduction technique is to save data only every s-th time step and use linear interpolation to approximate the missing values for offline data analysis. This technique can achieve arbitrary reduction ratios, but it lacks control over the errors. While we will support this technique, our data reduction goal is to preserve the essential information in the reduced output while satisfying resource constraints on I/O bandwidth. Thus, we need data reduction methods that provide control over errors.

Lossy data reduction is already used in consumer environment and the consumer big data domain is in advance of science in the systematic use of lossy data reduction. Most photos taken on a smartphone are stored in lossy compressed form, as are audio and video files. The projection made by CISCO about the Internet traffic is striking: in 2025, 80% of the Internet traffic will be video streaming; which means that more than 80% of the data transiting on the Internet will be lossy compressed. Microsoft has already deployed FPGAs into its data centers to accelerate JPEG compression (among other operations). An important distinction between the scientific and consumer big data domains is the specificity of the data reduction techniques. The consumer big data domain relies on generic lossy compressors (e.g., JPEG for images, MP3 for audio and MPEG4 for video). Many scientific applications at extreme scale already need aggressive data reduction. Spatial sampling and decimation in time are used to reduce data but these techniques also reduce significantly the quality of the data analytics performed on the sampled or decimated datasets. Advanced lossy compression techniques provide a solution to this problem by allowing the user to better control the data reduction error. However, the adoption of lossy data reduction techniques in the scientific domain is still limited because of the lack of comprehensive understanding of the errors introduced by lossy data reduction.

Although lossy data reduction is critical to evolve many scientific domains to the next step, the technology of scientific data reduction and the understanding on how to use it are still in their infancy. The first evidence is the lack of results in this domain: over the 26 years of the prestigious IEEE Data

Compression Conferences, only 12 papers identify an aspect of scientific data in their title (floating-point data, data from simulation, numerical data, scientific data). The second evidence is the poor data lossy reduction performance on some datasets. Beyond the research on data reduction techniques, scientists also need to understand how to use lossy data reduction. The classic features of compressors (integer data compression, floating-point data compression, fast compression and decompression, error bounds for lossy compressors) do not characterize data reduction algorithms specifically with respect to their integration into a high-performance computing and data analytics workflow.

The ECP CODAR codesign project is addressing these two gaps by collecting data reduction need from exascale application, investigating and developing new lossy data reduction algorithms, collecting error assessment needs from applications and developing a tool, called Z-checker, to assess comprehensively the error introduced by lossy data reduction.

Lossy scientific data reduction can be done following different approaches. A first method is to ask application and system developers to design lossy data reduction technique specific to the application. A good example the LHC, which already reduces the data produced by the detectors and plans to reduce the data even more for the run 3 [4]. The raw data per event is about 1MB for Atlas and CMS, and 100KB for LHCb. Atlas, CMS and LHCb currently produces events respectively at 100Mhz and 1Ghz for run 2. Run 3 will produce events for these experiments respectively at 0.4Mhz, 0.5Khz and 40Mhz. These detectors will produce a gigantic amount of data at an extraordinary rate: 60TB/s for ATLAS and CMS, 2TB/s for LHCb. To tackle this unprecedented data flow, the Alice project has defined a new combined offline-online framework called O2 that supports data flows and processing. It performs online compression of events to reduce data rate to storage to 20 GB/s. For the run 3, the O2 framework design features 463 FPGAs detector for readout and fast cluster finding, 100,000 CPU cores to compress 1.1 TB/s data streams, 5000 GPUs to speed up the reconstruction. The different lossy data reduction method is to design and use generic lossy compressors for scientific data. Several teams have worked and are still working on this problem. The difficulty here is to develop lossy compressors that provide excellent data reduction performance for a large variety of scientific applications: regular mesh, irregular mesh, particle simulation, instrument, etc.

Appropriately chosen reduction methods can improve the information content of output data. For example, the FLASH **hydrodynamics** simulation code [15] is widely used to perform extremely large simulations. Conventionally, its data is output only every s-th time step, and the remaining data is discarded. An alternative curve-fitting technique exploits the fact that hydrodynamic flows are mostly smooth and thus can be greatly reduced by lossy compressors that nevertheless provide error bounds. Our SZ compressor [41], for example, can achieve 100:1 reduction for the BLAST2 hydrodynamics data [9].

Currently, the two leading lossy compressors for scientific data are SZ [42,46] and ZFP [49]. They are error-bounded lossy compressors, meaning that they respect user-specified error constraints. Each uses a completely different compression strategy. One is based on a prediction method and the other one is transform based. One is better than the other, depending on the application and the dataset. Research in this domain aims to reach compression factors of 10 for hard to compress data sets and >100 for easy to compress ones. These two lossy compressors as well as other generic lossy compressors for scientific data work well for smooth datasets. They are less effective when the data sets are very irregular and presents large variations. One important aspect of the CODAR project is to understand what compression algorithm (or sequence of algorithms) to use according to the characteristics of the datasets. We return to this question in the next section.

5.3 Monitoring services

Another important distinction between scientific and consumer big data domains is the difference in quality requirements concerning the reduced data set. JPEG, MP3 and MPEG4 are not only generic but universal: all users have the same perception of images and sound. Thus, compression quality criteria can be defined that meet the needs of a large population of users. In science, on the other hand, each combination of application and data may involve different quality requirements. One open question is the relevant set of quality criteria for scientific data sets. Users have already expressed need to assess spectral alteration, correlation alteration, the statistical properties of the compression error, the alteration of first and second order derivatives and more. As the domain of lossy data reduction for scientific data sets grown, the community will learn what metrics are relevant and needed.

Another open question is how to express the quality requirements, in particular when these requirements are in large number with and dependencies. Perhaps the most important open question is the comprehensive assessment of the error introduced by lossy data reduction. The classic lossy compressor assessment metrics, PSNR (peak signal to noise ratio) and its extension, the rate distortion diagram, are not enough to represent the potential impact of the error on the scientific data sets and the analysis that will be performed from them. Users are also interested by other distortions (spectral, derivative, distribution) and other characterization of the error (autocorrelation, distribution). The ECP CODAR project is developing the Z-checker software framework to provide comprehensive assessment of the initial data set properties and of the alterations introduced by lossy data reduction. Z-checker is designed as a community tool and can integrate analysis modules in C, C++, Fortran and R. Z-checker itself will evolve as a parallel application to run many analysis concurrently and produce automatically an assessment report covering user specified analysis.

As illustrated in Figure 2, blindly applying a reduction method can result in a failure to capture features that are essential for offline analysis. Our data monitoring services will provide mechanisms to estimate data reduction errors and provide (1) feedback to the reduction methods so that their tolerances can be adjusted and (2) reduction error maps for the application scientist. These maps can be imported into offline data analysis routines or visualized to observe the evolution of reduction errors.

We will first provide a simple monitoring service that computes the data reduction error by finding the difference between the actual simulation output or data analyses performed on that output (the ground truth) and the same data produced using the reduced simulation output (an approximation to the ground truth). If the ground truth and its approximation are close, then we conclude that the data reduction is behaving as expected. If they are not close, then we will provide refinement hints to the CODAR Data Reduction Services that can include where the data was reduced either too aggressively or not aggressively enough. A key to monitoring is computing the analysis and difference by using the reduced data representation without doubling the memory footprint of the simulation by reconstructing the full set of outputs. As monitoring becomes more important to applications, additional application-specific monitoring services will be introduced.

The difference between the ground truth and its approximation will be computed by using a metric. Initially, we will use simple pointwise metrics (e.g., L_1, L_2, and L_∞) and statistical metrics (e.g., Kullback-Leibler divergence) when comparing distributions. When possible, we will use the natural metrics specific to the data analysis (e.g., bottleneck and Wasserstein distances in topological analysis [25]).

Important questions to be answered include the following. How frequently should we estimate the reduction error? What data analysis methods and metrics should we use for this estimation? How quickly can we provide the refinement hints so that the information provided is actionable? How effective are the refinement hints at influencing the reduction error?

Z-checker is our tool to explore the features of scientific data sets and understand the data alteration after (lossy) compression in a systematic and reliable way. Z-checker combines a battery of data analysis components relevant for data compression in an open-source tool for which users and developers can contribute and add new analysis components based on their additional analysis demand. Currently, Z-checker can be used to characterize critical properties (such as entropy, distribution, power spectrum, principle component analysis, auto-correlation) of any data set to improve compression strategies, detect the compression quality (compression ratio, bit-rate), and provide global distortion analysis comparing the original data with the decompressed data (peak signal-to-noise ratio, normalized mean square error , rate-distortion, rate-compression error, spectral, distribution, derivatives) and statistical analysis of the compression error (maximum/minimum/average error, autocorrelation, distribution of errors).

6 Related work

The growing disparity between compute and I/O rates has spurred much work on high-performance online (also termed, in the case of analyses, “in-situ” and “in-transit”) data analysis and reduction methods [5, 10], motivated by a desire to conserve I/O bandwidth, storage, and/or power; increase accuracy of data analysis results; and/or make optimal use of parallel platforms [34], among other factors [3].

Iverson et al. [23].

The problem has spurred various science teams to create custom online data analysis and reduction techniques [18, 24, 29, 30, 39, 43]. Such work makes clear the importance of such methods for the successful use of exascale computers. It also reveals complex relationships between application design, data analysis and reduction methods, programming models, system software, hardware, and other elements of an exascale system, particularly given constraints such as applicability, fidelity, performance portability, and power efficiency.

However, the community is far from completely understanding the many co-design issues posed by online data analysis and reduction. Indeed, as noted in a recent DOE report many other science teams have not yet developed their own infrastructure and “newer applications will postpone such implementations.” [2] For the broader community to leverage and expand the knowledge gained by early adopters, they will require an effective, usable and sustainable software infrastructure that allows scientists to use the *best techniques* to extract the *right information* that can then be pushed through the straw to the parallel file system.

and also on general-purpose methods [11, 41]

7 Conclusion

TBD

Acknowledgments

This research was supported in part by the Exascale Computing Project (17-SC-20-SC) of the U.S. Department of Energy, and by the Advanced Scientific Research Office (ASCR) at the Department of Energy, under contract DE-AC02-06CH11357.

References

1. ITER. `http://www.iter.org`. Visited June 1, 2016.
2. Large scale computing and storage requirements for advanced scientific computing research: Target 2017. report of the HPC requirements review, 2014. `http://www.nersc.gov/assets/HPC-Requirements-for-Science/ASCR2017/ASCR2017Final.pdf`.
3. J. Ahrens. Increasing scientific data insights about exascale class simulations under power and storage constraints. *IEEE Computer Graphics and Applications*, 35(2):8–11, 2015.

4. J. Albrecht. Challenges for the lhc run 3: Computing and algorithms. *Presentation at International workshop on Advanced Computing and Analysis Techniques in physics research, Jan 2016, UTFSM, Valparaso (Chile)*, 2016.
5. A. C. Bauer, H. Abbasi, J. Ahrens, H. Childs, B. Geveci, S. Klasky, K. Moreland, P. O'Leary, V. Vishwanath, B. Whitlock, and E. W. Bethel. In situ methods, infrastructures, and applications on high performance computing platforms. *Computer Graphics Forum*, 2016.
6. A. Biswas, S. Dutta, H.-W. Shen, and J. Woodring. An information-aware framework for exploring multivariate data sets. *IEEE Transaction on Visualization and Computer Graphics*, 19(12):2683–2692, 2013.
7. P. Bonoli and L. C. McInnes. Report of the workshop on integrated simulations for magnetic fusion energy sciences, 2015. `https://www.burningplasma.org/resources/ref/Workshops2015/IS/ISFusionWorkshopReport.11.12.2015.pdf`.
8. D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete and Computational Geometry*, 37:103–120, 2007.
9. P. Colella and P. R. Woodward. The piecewise parabolic method (ppm) for gas-dynamical simulations. *Journal of Computational Physics*, 54(1):174–201, 1984.
10. M. Dorier, M. Dreher, T. Peterka, J. M. Wozniak, G. Antoniu, and B. Raffin. Lessons learned from building in situ coupling frameworks. In *1st Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization*, pages 19–24. ACM, 2015.
11. M. Dreher and B. Raffin. A flexible framework for asynchronous in situ and in transit analytics for scientific simulations. In *14th International Symposium on Cluster, Cloud and Grid Computing*, pages 277–286. IEEE, 2014.
12. H. Edelsbrunner and J. Harer. Persistent homology — a survey. In *Surveys on Discrete and Computational Geometry. Twenty Years Later.*, volume 453 of *Contemporary Mathematics*, pages 257–282. American Mathematical Society, 2008.
13. H. Edelsbrunner and D. Morozov. Persistent homology: Theory and practice. In *Proceedings of the European Congress of Mathematics*, 2012.
14. I. Foster, D. R. Kohr Jr, R. Krishnaiyer, and A. Choudhary. Double standards: Bringing task parallelism to HPF via the Message Passing Interface. In *ACM/IEEE Conference on Supercomputing*, pages 36–36. IEEE, 1996.
15. B. Fryxell, K. Olson, P. Ricker, F. Timmes, M. Zingale, D. Lamb, P. MacNeice, R. Rosner, J. Truran, and H. Tufo. FLASH: An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. *The Astrophysical Journal Supplement Series*, 131(1):273, 2000.
16. H. Guo, W. He, T. Peterka, H.-W. Shen, S. Collis, and J. Helmus. Finite-time Lyanpunov exponents and Lagrangian coherent structures in uncertain unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 22(6):1672–1682, 2016.
17. H. Guo, W. He, S. Seo, H.-W. Shen, and T. Peterka. Extreme-scale stochastic particle tracing for uncertain unsteady flow analysis. In *Proceedings of Supercomputing 2016 (SC16)*, submitted 2016.
18. S. Habib, A. Pope, H. Finkel, N. Frontiere, K. Heitmann, D. Daniel, P. Fasel, V. Morozov, G. Zagaris, T. Peterka, et al. HACC: Simulating sky surveys on state-of-the-art supercomputing architectures. *New Astronomy*, 42:49–65, 2016.
19. S. Herbein, M. Matheny, M. Wezowicz, J. Krogel, J. Logan, J. Kim, S. Klasky, and M. Taufer. Performance impact of I/O on QMCPack simulations at the petascale and beyond. In *Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on*, pages 92–99. IEEE, 2013.

20. H. Huang, H. Qin, S. Yoo, and D. Yu. Local anomaly descriptor: A robust unsupervised algorithm for anomaly detection based on diffusion space. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 405–414, 2012.
21. H. Huang, H. Qin, S. Yoo, and D. Yu. A new anomaly detection algorithm based on quantum mechanics. In *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, pages 900–905, 2012.
22. H. Huang, H. Qin, S. Yoo, and D. Yu. Physics-based anomaly detection defined on manifold space. *TKDD*, 9(2):14:1–14:39, 2014.
23. J. Iverson, C. Kamath, and G. Karypis. Fast and effective lossy compression algorithms for scientific datasets. In *European Conference on Parallel Processing*, pages 843–856. Springer, 2012.
24. J. Jenkins, I. Arkatkar, S. Lakshminarasimhan, D. A. Boyuka II, E. R. Schendel, N. Shah, S. Ethier, C.-S. Chang, J. Chen, H. Kolla, et al. ALACRITY: Analytics-driven lossless data compression for rapid in-situ indexing, storing, and querying. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems X*, pages 95–114. Springer, 2013.
25. M. Kerbert, D. Morozov, and A. Nigmetov. Geometry helps to compare persistence diagrams. In *Proceedings of the Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2016.
26. Q. Koziol, N. Podhorszki, S. Klasky, Q. Liu, Y. Tian, M. Parashar, K. Schwan, M. Wolf, and S. Lakshminarasimhan. ADIOS. In *High Performance Parallel I/O*, pages 203–213. Chapman and Hall/CRC, 2014.
27. S. Ku, C. Chang, M. Adams, J. Cummings, F. Hinton, D. Keyes, S. Klasky, W. Lee, Z. Lin, S. Parker, et al. Gyrokinetic particle simulation of neoclassical transport in the pedestal/scrape-off region of a tokamak plasma. *Journal of Physics: Conference Series*, 46(1):87, 2006.
28. K. Kumaran. Introduction to Mira. `https://www.alcf.anl.gov/files/bgq-perfengr.pdf`. Visited June 20, 2016.
29. S. Lakshminarasimhan, J. Jenkins, I. Arkatkar, Z. Gong, H. Kolla, S.-H. Ku, S. Ethier, J. Chen, C. S. Chang, S. Klasky, R. Latham, R. Ross, and N. F. Samatova. ISABELA-QA: Query-driven analytics with ISABELA-compressed extreme-scale scientific data. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 31:1–31:11, New York, NY, USA, 2011. ACM.
30. S. Lakshminarasimhan, N. Shah, S. Ethier, S.-H. Ku, C.-S. Chang, S. Klasky, R. Latham, R. Ross, and N. F. Samatova. ISABELA for effective in situ compression of scientific data. *Concurrency and Computation: Practice and Experience*, 25(4):524–540, 2013.
31. Q. Liu, J. Logan, Y. Tian, H. Abbasi, N. Podhorszki, J. Y. Choi, S. Klasky, R. Tchoua, J. Lofstead, R. Oldfield, M. Parashar, N. Samatova, K. Schwan, A. Shoshani, M. Wolf, K. Wu, and W. Yu. Hello ADIOS: The challenges and lessons of developing leadership class I/O frameworks. *Concurrency and Computation: Practice and Experience*, 26(7):1453–1473, 2014.
32. X. Liu and H. Shen. Association analysis for visual exploration of multivariate scientific data sets. *IEEE Transaction on Visualization and Computer Graphics*, 22(1):955–964, 2016.
33. Z. Liu, B. Wang, T. Wang, Y. Tian, C. Xu, Y. Wang, W. Yu, C. A. Cruz, S. Zhou, T. Clune, et al. Profiling and improving I/O performance of a large-scale climate scientific application. In *22nd International Conference on Computer Communication and Networks (ICCCN)*, pages 1–7. IEEE, 2013.

34. P. Malakar, V. Vishwanath, T. Munson, C. Knight, M. Hereld, S. Leyffer, and M. E. Papka. Optimal scheduling of in-situ analysis for large-scale scientific simulations. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, page 52. ACM, 2015.
35. L. Nowell. Science at extreme scale: Architectural challenges and opportunities, 2014. `http://www.mcs.anl.gov/~hereld/doecgf2014/slides/ScienceAtExtremeScale_DOECGF_Nowell_140424v2.pdf`.
36. T. Perciano, D. Ushizima, E. W. Bethel, Y. D. Mizhahi, and J. A. Sethian. Reduced-complexity image segmentation under parallel markov random field formulation using graph partitioning. In *2016 IEEE International Conference on Image Processing*, page forthcoming. IEEE, 2016.
37. T. Perciano, D. Ushizima, E. W. Bethel, Y. D. Mizhahi, and J. A. Sethian. Reduced-complexity image segmentation under parallel Markov random field formulation using graph partitioning. In *IEEE International Conference on Image Processing*, 2016.
38. J. R. Perilla, B. C. Goh, C. K. Cassidy, B. Liu, R. C. Bernardi, T. Rudack, H. Yu, Z. Wu, and K. Schulten. Molecular dynamics simulations of large macromolecular complexes. *Current opinion in structural biology*, 31:64–74, 2015.
39. T. Peterka, J. Kwan, A. Pope, H. Finkel, K. Heitmann, S. Habib, J. Wang, and G. Zagaris. Meshing the universe: Integrating analysis in cosmological simulations. In *International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 186–195. IEEE, 2012.
40. T. Peterka, R. Ross, B. Nouanesengsey, T.-Y. Lee, H.-W. Shen, W. Kendall, and J. Huang. A study of parallel particle tracing for steady-state and time-varying flow fields. In *Proceedings of IPDPS 11*, Anchorage AK, 2011.
41. F. C. S. Di. Fast error-bounded lossy HPC data compression with SZ. In *IEEE International Parallel and Distributed Processing Symposium*, 2016.
42. F. C. S. Di. Fast error-bounded lossy HPC data compression with SZ. In *IEEE International Parallel and Distributed Processing Symposium*, 2016.
43. E. R. Schendel, Y. Jin, N. Shah, J. Chen, C. S. Chang, S. H. Ku, S. Ethier, S. Klasky, R. Latham, R. Ross, and N. F. Samatova. ISOBAR preconditioner for effective and high-throughput lossless data compression. In *2012 IEEE 28th International Conference on Data Engineering*, pages 138–149, April 2012.
44. A. Shekhar, K.-i. Nomura, R. K. Kalia, A. Nakano, and P. Vashishta. Nanobubble collapse on a silica surface in water: Billion-atom reactive molecular dynamics simulations. *Physical Review Letters*, 111(18):184503, 2013.
45. M. Slawinska, M. Clark, M. Wolf, T. Bode, H. Zou, P. Laguna, J. Logan, M. Kinsey, and S. Klasky. A Maya use case: Adaptable scientific workflows with ADIOS for general relativistic astrophysics. In *Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery*, page 54. ACM, 2013.
46. D. Tao, S. Di, Z. Chen, and F. Cappello. Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization. In *IEEE International Parallel and Distributed Processing Symposium*, 2017.
47. P. Thibodeau. Coming by 2023, an exascale supercomputer in the U.S. `http://spectrum.ieee.org/computing/hardware/when-will-we-have-an-exascale-supercomputer`. Visited June 20, 2016.
48. D. Ushizima, H. Bale, E. W. Bethel, P. Ercius, B. Helms, H. Krishnan, L. Grinberg, M. Haranczyk, A. Macdowell, K. Odziomek, D. Parkinson, T. Perciano, R. Ritchie, and C. Yang. SAIDE: Scaling analytics for image-based data from experiments. *Journal of the Minerals, Metals and Materials Society*, 2016. accepted.

49. P. Windstorm. Fixed-rate compressed floating-point arrays. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2674–2683, Dec 2014.
50. J. M. Wozniak, T. G. Armstrong, K. C. Maheshwari, D. S. Katz, M. Wilde, and I. T. Foster. Interlanguage parallel scripting for distributed-memory scientific computing. In *Proc. WORKS at SC*, 2015.
51. L. Wu, K. Wu, A. Sim, M. Churchill, J. Y. Choi, A. Stathopoulos, C. Chang, and S. Klasky. Towards real-time detection and tracking of blob-filaments in fusion plasma big data. *arXiv preprint arXiv:1505.03532*, 2015.
52. Y. Zhao, M. Wilde, and I. Foster. Virtual Data Language: A typed workflow notation for diversely structured scientific data. In I. Taylor, E. Deelman, D. Gannon, and M. Shields, editors, *Workflows for eScience*, pages 258–278. Springer, 2007.