Research Opportunities for Ensuring HPC Data Integrity and Provenance

Chunhua Liao[1], Matthew Sottile, Tristan Vanderbruggen, Steve Chapin.
Lawrence Livermore National Laboratory

Introduction: Verification of trustworthiness in computing often focuses on computations instead of the data itself. Even when data is considered, the focus is generally on the computations related to handling the data. Trustworthy science requires data verification to be treated as equally important as computations. Data integrity is the maintenance of, and the assurance of, data accuracy and consistency over its entire life-cycle. It is foundational to trust in models and their predictions. Aiming to prevent unintentional changes to information, data integrity is a critical aspect to any system that stores, processes, or retrieves data. Advances in data integrity verification are necessary to ensure that data is not tampered with, provenance is tracked through the full lifecycle of a scientific modeling activity, and the semantics of the data conform to the requirements of modeling and analysis methods.

Use cases: We identify two typical scenarios in which data integrity is highly relevant to High-Performance Computing (HPC). First of all, traditional scientific computing using supercomputers is a data-heavy process that depends on data integrity to allow researchers to produce trustworthy results. Secondly, as machine learning techniques are being widely deployed in HPC, large amounts of datasets and AI models are becoming available for sharing and reusing in the community. There is an urgent need for improving data integrity and provenance for both datasets and AI models. Robust techniques for ensuring data integrity and provenance are necessary to give stakeholders confidence that they can trust data made available.

We believe that data integrity can be approached in at least three aspects. First, we are concerned with well-formedness – does the data conform to the I/O requirements of tools. An example requirement is that an input data file should conform to a predefined data schema stored in a preferred standard file format (e.g., XML) while satisfying constraints. Second, is the data meaningful at the application domain level – do physical, numerical, and geometric properties assumed by the application hold? Third, what is the provenance of the data – can a simulation result or AI models be traced in a trustworthy and secure fashion back to the exact authors and software/hardware configuration used to generate it? All of these aspects mentioned above have obvious cybersecurity implications.

Challenges: While existing input validation techniques are well established to achieve the first goal of ensuring well-formedness of data, schemas often are not rich enough to capture domain

[1]Corresponding author (liao6@llnl.gov). This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-ABS-827739.

level semantics. For example, an application may impose geometric constraints on a mesh that require a validator to not only examine the representation of the mesh as a collection of integers and floating point numbers, but examine the higher-level mesh structure that the collection denotes. Such constraints are outside the reach of validators today due to a lacking specification mechanism for expressing domain level data constraints and a lack of tooling to check such specifications. Last but not the least, current workflows for generating scientific data do not automatically provide sufficient, verifiable provenance information.

Research Opportunities: The first promising research direction is compile-time, runtime and offline steps to automatically check data integrity-related assertions throughout the life cycle of data generation, storage and retrieval. New forms of assertions will be needed to express rich semantic constraints of high-level domain concepts. They may also involve properties of an entire dataset (beyond just at the record or individual array level). An example is if a conservation law holds (Lagrangian remap example). These assertions can form the basis for offline verification of formal theories of data consistency and integrity using external theorem provers and satisfiability solvers. A number of formal methods tools that allow users to work at the level of mathematical theories (e.g., the Lean or Coq theorem provers) provide the building blocks for writing mathematical assertions and theories, but to our knowledge no tool exists to connect them to data schemas.

HPC programming models have traditionally been focusing on performance, correctness, and energy efficiency. There is an opportunity to incorporate data integrity as a first-class citizen into the design goals of programming models. Initial language, compiler and runtime extensions integrating data integrity and provenance techniques can be experimented using directive-based programming models (such as OpenMP). Programming models must be able to manage high-level semantics associated with complex (and potentially distributed) data structure. Doing so will allow data semantics to be analyzed and verified in concert with computational semantics to support data integrity-related logic to co-exist in the presence of parallelism and compiler optimizations.

Finally, blockchain-like techniques should be exploited for improving data provenance. To detect data tampering, all data must be signed and verified with low overhead. We need to study scalable and multi-institutional mechanisms to allow researchers from all over the world to manage their digital keys. As data is automatically generated by software, we also need to explore the possibility of treating software as agents and grant independent signing power to running software instances. So software-generated data can automatically have built-in data integrity and provenance information to enable data forensics.