Holistic View of Composable Data Analysis: Insights From Software Frameworks for Extreme Scale Computing

A. Dubey, W. Bethel, Prabhat, J. Shalf, A. Shoshani, B. Van Straalen

February 12, 2014

Across the DOE, data analysis for the experimental facilities has become an even bigger exascale challenge than simulations. The rate of data production is growing faster than Moore's law and curating, analyzing and archiving this data presents exascale challenges of its own. However, many of the challenges posed by the complex data analysis workflows share many characteristics with those of petascale and above computations. They consist of many different stages of computation/analysis, with different algorithms and diverse requirements from hardware and the software stack. Often these requirements conflict with one another, where optimizing for one is detrimental to another. These many moving parts also don't always play well together. Therefore, It takes a combination of carefully thought out design, hard-nosed trade-offs and good orchestration to succeed with multi-stage workflows whether they are for computations for simulations, or for analysis.

The most successful codes in the simulation community have evolved techniques to develop their solvers, algorithms and framework architecture in ways that enable many permutations and combinations of components to compose different applications. This is usually achieved through a combination of factors such as: (1) separating concerns of numerics and algorithms from those of housekeeping and other infrastructure; (2) identifying reusable common components; (3) devising interfaces through which the components can interact with one another; (4) looking at both optimal and sub-optimal options for implementing individual components for better interoperability; and (5) providing a high level framework to facilitate composability. While the individual components for the big data may be different from those in simulation codes, the same general principles would be a good basis for data analysis infrastructure development. In the following paragraphs we expand on some of these topics.

The identification of common reusable components is at the heart of all good software design. There are motifs and algorithms that appear over and over again in computations, which provide a natural coarse division for forming components. Some situations may warrant further divisions within motifs and algorithms, and it may sometimes be necessary to have several instantiations of the same component (for example there may be different solvers for the same equation with different stability and accuracy characteristics that are desirable for different physical regimes). However, irrespective of their granularity, applications typically use a subset of components in any one instance, the combination may vary between instances. This could be thought of as similar to stages in data analysis, where different stages apply different algorithms and have different motifs. The justification for well designed in-

terfaces in the presence of heterogeneous interoperating components is well understood and needs no further elaboration. Similarly, separation of concerns is a well established software design principle when there are heterogeneous components, though it has acquired a whole new dimension and urgency now due to increased complexity and heterogeneity of emerging hardware architecture.

The consideration of suboptimal options for individual components is motivated by the need for interoperability among components. In devising a simulation campaign, it is important to take the holistic view of all components, their relative importance and their relative contribution to the overall cost of calculation. In simulations, for example, often all the solvers use a single data layout for overall performance even if that layout is less than optimal for each of them individually. This is because the cost of rearranging data may be more than the loss in performance by keeping the same layout. The trade off is in the interest of fastest time to solution for the overall application. Similarly, a less accurate and correspondingly less expensive method may sometimes provide a solution within acceptable accuracy bounds for one instance of an application. This is particularly useful, for example, in pathfinder simulations where understanding the trend is more valuable than absolute values.

Orchestration becomes necessary whenever there are many heterogeneous components in a system. Here, it can even be leveraged to combine aspects of simulation with analytics. In particular, in-situ data analysis can play an important role in quantifying the trade-offs and the best ways of acting upon them. For example refining in AMR is an already exercised in-situ analysis that dictates the data granularity of various regions in the domain. It can also be used to steer simulations through in-flight parameter adjustment to minimize wasteful calculations. Additionally, it can facilitate the differentiation between useful and non-useful simulation data output, not only reducing the quantity of output data, but also providing ways of better data curation.

In addition to leveraging the acquired wisdom from the simulation communities, and cooperation through in-situ analysis there are other ways in which big data analysis and simulations meet. Experimental validation and design through simulations is one area where big data plays the role of intermediary. An example is a collaboration between the Flash Center and Oxford University for the laser shock plasma experiments, where the hypothesis of the experiment is modeled and simulated. There are two streams of data, one coming from the instruments, and the other from simulations. Each stream requires its own interpretation and analysis before the results of one can be confronted with those of the other. This approach is gaining increasing traction in experimental communities, where modeling the hypothesis of the experiment helps not only to enhance the reliability and credibility of the scientific results, but also serves the cause of improving the experiment design. Here, the workflow for the overall project includes simulations, experiments and data analysis from both in a closed loop for drawing scientific inferences.

Overall, the architecture and software engineering for large scale scientific data analysis can and should leverage many insights gained in extreme scale scientific computing. The methodologies employed in both spaces have a common set of principles that are essential for overcoming the complexity of our current hardware and software environments, and ensure a productive programming environment for the intimately related requirements of data analysis and simulations.