Big Data Analytics and High Performance Computing Convergence Through Workflows and Virtualization

Ewa Deelman
University of Southern California, Information Sciences Institute
Contact: deelman@isi.edu

There is a clear desire on the part of a number of domain and computer scientists to see a convergence between the two high-level types of computing systems, software, and applications: Big Data Analytics (BDA) and High Performance Computing (HPC). In some sense scientists do not care how they obtain their computational results, as long as they are generated in a timely and robust manner. Some applications naturally cross the boundaries between HPC and BDA, for example executing a large-scale simulation on an HPC system and then performing analytics on the results (either ex-situ—on another computational platform, or in-situ—within the HPC system.

Up to now, workflow management solutions have been developed to help cross the computational system boundaries. Workflows can be composed of a variety of computational tasks: tightly coupled codes, machine learning loosely coupled applications, and independent high-throughput tasks. Given a workflow description, the workflow management system can then select the appropriate resources, schedule the needed data movement, and send tasks for execution on the target resources. However, this solution keeps the different infrastructures separate and makes it hard to co-locate extreme computation and analytics.

Part of the problem in the convergence of BDA and HPC is the issue of BDA software deployment in HPC systems and the issue of performance in BDA architectures such as clouds (at least in terms of low latency communications). HPC systems are administered by entities such as campus IT infrastructures or National Lab IT personnel, so any changes to the software environment needs to be approved. Additionally, policies in place on HPC systems prefer tightly coupled applications over loosely coupled or embarrassingly parallel codes. On the other hand, clouds, which are the most common BDA platform often do not provide the high-performance networks at a scale needed by tightly coupled applications.

One possible solution is to support virtualization on the HPC systems, where the user (or workflow system or resource provisioner) could request a set of resources from the HPC scheduler and then manage these resources during application execution. The management would include the set up of the software environment on the resources, the scheduling of tasks onto these resources, monitoring, failure management, etc.

Unfortunately there are a number of drawbacks of this solution: 1) concerns over security of the HPC system, or concerns of using the system for malicious attacks, 2) complexity of setting up the correct software environment, 3) the complexity of the HPC system, in particular the deep memory hierarchy and its impact on the overall system energy consumption, and 4) potential performance degradation and suboptimal use of resources.

Some solutions to these problems could be:

1. Work closely with resource providers to understand concerns, develop “trusted” resource management systems, develop specialized monitoring tools, and auditing mechanisms.
2. Develop tools that automate the software environment set up, this would also need to include testing of the environment.

3. Develop data management capabilities that can seamlessly manage different types and amounts of data on behalf of the applications and workflow/resource management systems, make sure the data management capabilities provide an adequate level of abstraction and are easy to incorporate in legacy applications.
4. Realize that there may need to be some performance degradation in order to support scientific productivity and system manageability, develop tools that monitor resource usage and "penalize" applications and systems that waste unreasonable resources.

Finally, an important consideration regardless of the infrastructure and applications that users are executing, the systems need to be made reproducibility aware. They need to provide some level of insight into how reproducible the computation is, how the computation was performed (transparency) so that the results can be inspected, and how the environment and the applications were set up so that someone else can reproduce the results or reuse the methods or data.