January 12, 2015

The Role of Scientific Workflows in Bridging Big Data and Extreme Scale Computing

Ewa Deelman, University of Southern California
deelman@isi.edu

Workflow technologies have been demonstrated to be effective in exploiting coarse grain parallelism for distributed infrastructures such as grids and clouds. Up to now, workflow management systems have lived outside of the HPC systems, coordinating data movement in the wide area networks, data registration in catalogs, and sending jobs to the HPC systems through the local HPC schedulers. As the complexity of the applications and the HPC systems increases, workflow management systems need to take greater ownership of the computations being done inside the exascale machines. Where traditional job schedulers have considered CPU and core availability when scheduling jobs, workflow management systems can take into consideration data locality, data dependencies, and future application needs when scheduling complex workloads, composed of large numbers of inter-dependent tasks and workflow ensembles.

In our work, we made some initial inroads in moving the workflow management system (WMS) into the HPC systems. In particular, the Pegasus WMS[1] takes the approach of taking a high-level (abstract) workflow and mapping it onto the available computational, storage, and network resources. Once the workflow is mapped (compiled) onto the target system(s), the workflow engine executes the workflow. It stages data to and from the execution systems, sends individual tasks or sub-workflows to the target systems, monitors their execution, and addresses failures.

In some cases, individual tasks are managed by a local HPC or HTC scheduler, or more often especially in the case of HPC systems, sub-workflows, containing a number of inter-dependent tasks are being managed by a specialized execution engine. We have developed and experimented with an MPI-based, master-worker workflow execution engine. This resulted in our ability to run large-scale workflows on complex Cray XT architectures, where the nodes have a very minimal kernel with almost no system tools available, no shared libraries and limited IPv4/6 networking support. The networking on these nodes is usually limited to system-local connections over a custom, vendor-specific interconnect, making it hard to use existing workflow execution optimization solutions such as glide-ins.

Running tasks using an MPI-based engine still does not get to the heart of the problem of dealing with deep memory hierarchies available on today's and future extreme scale systems. One approach would be to design an interface between the workflow management system inside the HPC system (in-situ) and outside of it (ex-situ) and the compilers and runtime systems on the HPC machines. It would be interesting to explore the interplay between the integrated compiler and runtime system that work at a fine computational-granularity (intra-task) level and the workflow management system (WMS) that works at a coarser granularity (inter-task) level. By combining these systems in an intelligent way, we can hope to improve data locality within the HPC system and as a consequence optimize execution time and energy consumption.

Task execution: Application tasks provide a natural interface between workflow manager and the compiler/runtime systems. The compiler can identify the computational tasks within an application and expose them to the workflow management system (WMS) that is then responsible for scheduling them onto a set of nodes as part of an extreme-scale architecture and to manage all data transfers among tasks through shared memory or via message passing. The runtime system can then be responsible for sub-task-level scheduling and local memory management.

Data Management: The workflow management system provides a natural interface between the wide area data sources that store application input and output data and the extreme-scale system where the computations or visualizations take place. Traditionally, WMS have decided which data replica to access and how to lay out the files in a file system. The WMS also keeps track of which data was accessed,

[1] Pegasus Workflow Management System http://pegasus.isi.edu

which data was generated and how. As the WMS moves inside the HPC systems, it needs to add functionality to lay out the data for the application. It also needs to keep track of the coarse data movement within the system so that tasks can be scheduled in an energy-efficient way. With the goal of in-memory data sharing, co-location of data objects with each other and co-scheduling of tasks near the data they use, new in-situ data management techniques that are integrated with the WMS will need to be developed.

Tasks Scheduling: In this model, the workflow system will manage the storage and movement of data in distributed memory and schedule tasks "close" to the data locations. This will require new solutions for locating and indexing data in distributed memory, and for associating data with computational tasks that do not rely on file names and paths. There is also a potential need for new scheduling techniques that optimize data locality to make efficient use of deep memory hierarchies, and leverage high-level descriptions of the workflow structure to minimize data movement.

Reliability: Big Data workflow management systems deal with a number of failures present in workflow execution: task failures, problems accessing data, resource failures, and others. In the Big Data/Extreme-scale Computing context, it would also be worthwhile to investigate how data replication techniques can be used to improve fault tolerance when data is located in volatile memory as well as mechanisms for task-graph-based data-recalculation in case of task failures. In general, one can explore tradeoffs between data re-computation and data retrieval, taking into account time to solution and energy consumption. At the same time, one can also examine issues of staging data no longer needed off the extreme-scale system, while the computations are progressing.

Workflow Performance/Behavior Modeling: In order to be able to make decisions about task and data placement both across the wide area and inside an extreme-scale system, it is important to understand the resource needs and behavior of the workflow applications. This understanding needs to be across scales: from the level of a workflow ensemble, workflow instance, down to individual tasks and code segments. Much work has been done in modeling the low-level behavior of the applications. Recently, research is being conducted in modeling the resources needs of workflows (dV/dT Project[2]) and developing end-to-end, predictive, analytical models of workflow application performance (Panorama Project[3]).

Provenance Capture and Reproducibility Workflow Management Systems have been capturing provenance information about the creation, planning, and execution of workflow-based applications. Up to now, the approach has been to save everything, from the low-level details of the execution (OS, environmental variables, executable and its parameters, data used, etc.) up to the high-level workflow description, and even workflow composition steps—the workflow evolution—as the user is composing the workflow. When moving to extreme-scales it is impossible to save every piece of provenance information, since this information only adds to the storage requirements of the application, to the data movement inside the machine, and thus to the overall energy consumption. Thus, issues of the defining the level of detail that is most effective and efficient for capturing provenance information are very important. Provenance capture may need to adapt to the behavior of the application, perhaps collecting coarse granularity provenance most of the time, and capturing a finer-level of detail when interesting events are detected. If scientists find that the lack of some provenance information prevents accurate interpretation of the results, the system may want to automatically re-run parts of the computation and re-produce the results and a more detailed provenance trail.

---

[2] dV/dT: Accelerating the Rate of Progress Towards Extreme Scale Collaborative Science, https://sites.google.com/site/acceleratingexascale/

[3] Panorama: Predictive Modeling and Diagnostic Monitoring of Extreme Science Workflows, https://sites.google.com/site/panoramaofworkflows/home