Data Management in the Cloud: Let's Not Toss Out What Little Progress Towards Reproducibility We Have

Beth Plale	*Director, Data To Insight Center* *Associate Professor of Computer Science*
919 E. 10th St *School of Informatics and Computing* *Indiana University* *Bloomington, IN 47408*	*+1 (812) 855-4373 (voice)* *+1 (812) 856-3524 (alternate voice contact at Innovation Center)*
	plale@cs.indiana.edu

Topics of interest (from list, select 2):

1. Data portability, consistency, and management
4. Programming models for the cloud

Research interests in cloud computing

For the digital data created as an outcome of scientific discovery to retain its value over time, the data must undergo some level of curation. Curation is typically a human process of interacting with scientists to understand and capture conditions under which experiments were done. For archival of scientific data to be fully realized, however, curation costs must come down. Cloud computing raises the specter of losing the little bit of reproducibility science has achieved because of the unknown nature of the underlying platform. My lab has an active research project in provenance capture in cloud environments. Data provenance is the lineage record of a data object or data collection. We are interested in applications that execute in large-scale distributed settings such as GENI or a cloud setting, and provenance collection carried out at the middleware layer. We recently applied provenance capture in the Global Environments for Network Innovations (GENI) platform, specifically on PlanetLab. We apply provenance capture to an experiment running on GENI network and capture provenance information, including slice creation, topology of the slice and operational status of the experiment.

Specifically, we focused on provenance collection from the Gush shell script and execution management system to capture provenance information available at the experiment level about a particular experiment and its execution. To facilitate testing of the system, we used Twister [31], a parallel, iterative version of the MapReduce to execute a crawling application using breath-first search through a large-scale random graph. By utilizing MapReduce programming framework, the application explores the nodes of the same level of the graph in parallel, and then goes to the nodes in next levels iteratively. This way, it is able to process breadth-first graph search in parallel. This experiment was executed with GUSH and provenance of both successful and failed executions captured.

Instrumentation of GUSH required a new version of the Adaptor instrumentation type. The GENI adaptor provides an interface that uses the GENI experiment log files and a set of rules to derive provenance information and maps them into the Karma repository. The adaptor is simply a generic log processing unit for GENI log files, which comprise of two sub-units: Log Parser, Notification Generator. The Log Parser module is used to process log files to extract provenance information, while the Notification Generator is used for generating and sending provenance notifications to Karma repository. We utilized PlanetLab, where we run Gush to deploy and execute the Twister experiment. Gush requires that users describe their experiments or computation in an XML document. It uses this document to locate and access the remote

resources in PlanetLab. In its execution flow, Gush contacts a host to deploy a twister server, which then reads a configuration file and internally connects to other hosts where the application needs to run. One limitation of instrumenting Gush is that it does not capture the activity of the Twister server as it distributes jobs, schedules and executes them.

LEAD-in-a-Box is the next generation of LEAD, a self-contained system of climate and atmospheric analysis tools in a single multi-core server box that includes both web based and desktop clients, and supports workflow execution and collaborative sharing tools. It utilizes the Trident Scientific Workflow Workbench and Windows HPC Server. It consults public data sources in real-time such as at UCAR and NOAA for data used in analysis, and, when resource demands of the server box are exceeded, utilizes cloud resources on the back end in a way transparent to the user (except for the cost model). A key component in our suite of tools is Sigiri (Chinthaka et al. 2009), a resource management tool for deploying jobs (i.e., tasks or subworkflows) to heterogeneous platforms. It is designed to extend to new job descriptions languages and currently supports JSDL (Job Submission Description Language) and RSL (Resource Specification Language). A Trident activity designed to interact with Sigiri enables scientists to submit jobs to different computational resources within a Trident workflow. Once the required computational resource is selected (either by the user or by a quality of service optimization algorithm) this activity will pass this information, together with job descriptions and credentials, to Sigiri which will use appropriate daemon to submit this job to the computational resource selected. This activity can also be used to continuously monitor the progress of the submitted job, using the Sigiri Web services API, and report the state transitions. In an HPDC 2011 submission we utilize Sigiri to submit and monitor workflow tasks to Amazon EC2.

One page abstract of important future research problems

I remain captivated by the topic of data provenance capture for purposes of data preservation. As mentioned, for the digital data created as an outcome of scientific discovery to retain its value over time, the data must undergo some level of curation. This will be achieved in part through tools that automate metadata and provenance collection. Provenance is an important piece of the curation record. It traces the derivation history of a data product or collection of products. Provenance can identify event causality, can be used to attribute ownership and determine the quality of a particular data set; can enable broader forms of sharing, reuse, and long-term preservation of scientific data. Computing in the cloud amplifies the need for understanding the behavior and ultimate derivation of a data product because of the additional factors such as failures and latencies that come into play. Our first step in this direction is to add provenance instrumentation to the iterative version of MapReduce called Twister that is currently supported as middleware on FutureGrid. We use Cytoscape to visualize the provenance; it is suitable for large-scale graphs and we are customizing it for provenance.

A large set of environmental applications rely on real time observational data, and indeed many are developed to ingest continuous streams. The cloud will need new models to support the execution of non-persistent applications whose purpose is the processing data streams. Data objects, perhaps as stream snippets, need to flow as freely in a cloud data layer as are blocks replicated in a Google File System. For this problem we would begin as a starting point with our work in StreamFlow, a framework that provides a uniform eScience workflow semantics over scientific workflow composition and complex event processing.