Data Mining Via Smart Grid Workflow: The SciFlo Dataflow Execution Network

Brian Wilson, Dominic Mazzoni, Gerald Manipon, and Benyang Tang
Jet Propulsion Laboratory

The market penetration of Grid computing is being revolutionized by the new Service Oriented Architecture (SOA) paradigm in which remotely-callable services are published, discovered, and chained together over the Web. In standard Web Services (WS), remote procedure calls (RPC) are performed by exchanging XML messages over a standard communications protocol (usually Simple Object Access Protocol or SOAP), service interfaces are described in Web Service Description Language (WSDL), and the available services are published in searchable Universal Description, Discovery and Integration (UDDI) registries. The simplicity and growing market dominance of the WS (XML/SOAP/WSDL) software stack has in turn led to the development of standard service choreography or **Grid workflow** tools, such as Business Process Execution Language (BPEL) in the commercial world and multiple contenders in the scientific world (Pegasus, Kepler, Triana, SciFlo, etc.). Layered on top of the services paradigm is the concept of the "Grid", in which compute clusters and storage resources are discovered on the Grid, allocated by authenticated users, used, and then released, with remote usage controlled by permissions and policy, and backed by full auditing and billing. All of these capabilities are available in the latest, or web services, version of the Globus toolkit, which is now commercially supported.

In this new generation of the Web/Grid, it is now simple to **publish algorithms for reuse** as services on the web. In fact, one could argue that publishing Web (SOAP) services represents the first, large-scale opportunity in the history of computing to reuse software in a fine-grained way and to enable easily reconfigurable, machine-to-machine, automated data processing. This opportunity should not be underestimated. A few years from now, any algorithm that is not available as a discoverable & machine-callable service on the web will simply not exist in customer's minds, just as projects without web pages, individuals without home pages or blogs, or science papers without on-line preprints won't really exist. The data mining community must be cognizant of this trend and exploit it. Over the long history of data mining, a variety of software frameworks have been developed in which mining operators and machine learning algorithms are plugged into a framework or pipeline. Unfortunately, the adoption and lifetime of such frameworks has always been limited by changing "fads" in programming language popularity (C++, Java, C#) and software component & RPC technologies (CORBA, COM, Java RMI). We argue that the simplicity and universality of XML messaging represents a quantum leap forward in component and RPC technologies, and the SOA/Grid paradigm provides a near-universal framework that will have a longer lifetime and is thus worth supporting.

SciFlo stands for **Sci**entific data**Flo**w. The SciFlo Web is a network of Semantically-Enabled or "Smart" Dataflow Execution nodes. SciFlo leverages Web (SOAP) Services and the Grid Computing standards (WS-Resource Framework, other WS-* standards & the Globus toolkit), and enables scientists to do multi-sensor Earth System Science by assembling reusable web services, python-wrapped operators, command-line scripts, and native executables into a distributed computing flow (tree of operators). The SciFlo parallel dataflow engine was developed to enable large-scale, multi-instrument atmospheric science using combined datasets from NASA's AIRS, MODIS, MISR, and GPS sensors. Investigations include cross-comparison of spaceborne climate sensors, cloud spectral analysis, study of upper troposphere-stratosphere water transport, study of the aerosol indirect cloud effect, and global climate model validation. The challenges are to bring together very large datasets, reformat and understand the individual instrument retrievals, co-register or re-grid the retrieved physical parameters, perform computationally-intensive data fusion and data mining operations, and accumulate complex statistics over months to years of data. To meet these challenges, we are deploying SciFlo nodes at several NASA data centers (JPL, Goddard, Langley) and multiple universities, and a set of versatile and reusable operators for data access, subsetting, registration, mining, fusion, compression, and advanced statistical analysis.

The SciFlo client & server engines optimize the execution of distributed data flows and allow the user to transparently find and use datasets and operators without worrying about the actual location of the Grid resources. The scientist injects a distributed computation into the Grid by simply filling out an HTML form or directly authoring the underlying XML dataflow document, and results are returned directly to the scientist's desktop. Once an analysis has been specified for a chunk or day of data, it can be easily repeated with different control parameters or over months of data.

There are serious performance issues inherent in using XML messaging for distributed computing and XML representations of large scientific datasets. SciFlo addresses these issues in several ways. First, in the data mining domain, operators (algorithms) are often computationally intensive and operator pipelines are fairly coarse grained. Thus, typical workflows will not be making hundreds or thousands of web service (or operator) calls per second, so the overhead of the XML messaging engine is not a concern. Performance will be limited by the time taken in executing operators and moving data between operators. Second, XML formats are only used in SciFlo: to specify the workflow, to control the workflow execution channel, and to exchange metadata. Large scientific datasets, such as instrument scans, images or 3D/4D variable grids, are kept in standard binary containers (formats appropriate for the science domain, e.g., netCDF and HDF) and "referred to" in the workflow using ftp or http URL's. Each SciFlo node also contains an Open Data Access Protocol (OpenDAP) server. The OpenDAP protocol (an http URL with a query part) allows one to "drill down" inside a remote netCDF or HDF file and fetch a slice of any variable grid. This fine-grained, file subsetting capability can often be used to improve performance since grabbing only the needed variables or grid slices from a file is more efficient than moving the entire file and then locally subsetting.

In SciFlo, the user never explicitly asks for or "programs" data movement. All data is referred to via URI/URL's and movement is implicit. The SciFlo engine automatically and transparently moves only the data needed to support operator execution. SciFlo also tries to minimize data movement by supporting two styles of distributed computing: one can both move data to the operators, and move operators to the data. For example, a data mining operator that needs to run over a large dataset can be pushed into a SciFlo execution node within that data center. Or a data fusion workflow that requires one large dataset and a smaller dataset can be executed at a node "near" the large dataset so only the smaller dataset is accessed over the Internet backbone.

SciFlo data mining or fusion operators are (if possible) implemented as incremental or "streaming" operators that accumulate aggregates or statistics (e.g. grid averages) on the fly. The operators are initialized (accumulators started), a stream of data or list of data files is sequentially presented via the workflow, and then the final statistics or aggregations are computed. (The data are streamed through the operator tree.) Long operator pipelines, consisting of services chained together over the web, represent a serious performance challenge due to the overhead in repeatedly reading from & writing data to binary files and moving the bits over the Internet. If a set of operators are movable, then they should be consolidated at a single location and preferably chained together with "in memory" data streaming. If the service chain is inherently distributed but the operators are streaming-enabled (and read a data stream from a socket), the data bits can be moved using a specialized streaming channel (protocol) rather than incurring the overhead of disk access and file transfer via the ftp or http protocols. Even if the data are streamed, the workflow can still be assembled and the operators prepared using the XML/SOAP control channel. Thus, SciFlo can be thought of as a multi-channel data processing system in which workflow assembly and execution control occur on one (ASCII) XML channel, while file transfer or data streaming occurs on one or more additional binary channels. The binary channels can use standard protocols to move data chunks such as ftp, http or OpenDAP (support for these is built into SciFlo), or proprietary protocols for streaming data, or high-speed bulk transfer over optical fibers (LambdaRail protocols).

In our presentation, we will discuss the design issues and solutions used in the implementation of SciFlo, including: the design of XML dataflow documents, use of XML formats for metadata, implementation of the parallel dataflow execution engine, design of reusable operators & web services, and discovery of suitable remote services that have been published by others. To illustrate the SciFlo concepts, several data mining and data fusion examples will be discussed and then executed live. Performance comparisons for different execution scenarios (with & without operator movement) will also be presented. Finally, we will discuss the prospects for adding data streaming support to the SciFlo execution engine.