Service-Oriented Architecture for Biosequence Analysis Workflow

Adam Hughes[*], Saliya Ekanayake[*†], Judy Qiu[*†], and Geoffrey Fox[*†]
[*]Pervasive Technology Institute
[†]School of Informatics and Computing
Indiana University, Bloomington IN, USA
{adalhugh, sekanaya, xqiu, gcf@indiana.edu}

Abstract— **The advent and continued refinement of modern high-throughput sequencing techniques have led to a proliferation of raw biosequence data, as labs routinely generate millions of sequence reads in a matter of days. Analyzing these results is beyond the computational capacity of single-lab resources, necessitating the use of high-performance computing resources, which presents significant overhead in terms of manual data manipulation and job monitoring. To overcome this bottleneck, application pipelines are being developed in conjunction with a generalized service-based portal system to abstract the details of job creation and management. This infrastructure will eventually provide service interfaces to map-reduce technologies, leveraging the computing resources offered by traditional grids and clusters, as well as emerging Cloud platforms.**

This paper first presents the design and implementation of a software package to automate the setup steps for a specific sequence analysis workflow. Using this workflow system as a foundational example, the design and implementation of a generalized service-oriented architecture (SOA) to support the creation and management of biosequence analysis jobs is discussed. Finally, plans to simplify the service-creation process and to provide access to a hybrid computing environment of grids, clusters, and Cloud platforms are presented.

Keywords-biosequence; bioinformatics; SOA; service-oriented architecture

I. INTRODUCTION

Genome research projects generate large quantities of data in the form of biosequence reads, each consisting of several hundred base pairs. In order to derive useful information from these data, scientists employ many software tools to search generated biosequences against existing databases, to visualize these search results, and to create and manage biosequence annotations. The sheer size of the experimental data, which may consist of hundreds of millions of sequence reads generated from a single run (as discussed in [1] and [2]), renders the computational resources of single labs inadequate for completing such analyses in a timely fashion. Consequently, the use high-performance computing resources is generally required to make progress against the avalanche of sequencing results facing genomic researchers. Creating and managing the computational jobs necessary to carry out these analyses can be a tedious manual task, potentially costing the researcher valuable time and effort. In order to overcome this productivity bottleneck, robust application workflows for automating most of the steps involved in sequence analysis and a service-based portal system to abstract the intricacies of managing interactions with underlying high-performance computing machines are being developed. This infrastructure will be expanded to provide service interfaces to map-reduce technologies and make the computing resources offered by traditional grids and clusters, as well as emerging Cloud platforms, more accessible to typical sequencing projects.

This paper first presents the design and implementation of a software package to automate the setup steps for a specific sequence analysis workflow. Using this workflow system as a foundational example, the design and implementation of a generalized service-oriented architecture (SOA) to support the creation and management of biosequence analysis jobs is discussed. Finally, plans to simplify the service-creation process and to provide access to a hybrid computing environment of grids, clusters, and Cloud platforms are presented.

.

II. RELATED WORK

Providing ready access to high-performance computing resources for biological researchers has been the focus of system engineering research for many years. Various portal systems have been developed to provide users with a clean interface for submitting and controlling HPC jobs [3]. Many of these implementations, though, have been tied to specific platforms or a limited number of distributed computing environments, such as Condor or SGE [4], generally lacking seamless access to Cloud and other other HPC resources. The service-based technology of the Portal system described here is a further abstraction of these earlier ideas, providing simple interfaces that allow a wide range of applications to access potentially any type of computing resource, from local clusters to Cloud platforms through flexible resource connector components. The lightweight, loosely-coupled architecture [12] of these services makes

them readily accessible for quickly composing domain-specific software to address problem spaces of interest, thus providing the ability to quickly construct new distributed analysis tools.

In the realm of biosequence analysis, many computational pipelines have been developed to assist in genome annotation [5]. A common early step in such workflows is sequence alignment, wherein a set of input sequences are compared to a databases of well-known sequences, such as the human genome, or against each other. One important tool for performing this type of analysis is the Smith Waterman-Gotoh (SW-G) algorithm, a local sequence-alignment tool which calculates similarities between regions of protein or DNA sequences and scales like O(N2) [6, 7]. Recent work has focused on the use of an MPI version of SW-G to examine Alu clustering[8,9,10], a particularly vexing problem since Alu represents the largest repeating families in the human genome[9,10]. This implementation has proven valuable for efficient Alu clustering on shared-memory multicore systems and distributed-memory clusters. In addition to the MPI version of Smith Waterman-Gotoh, multiple implementations have been developed with an eye toward deployment to Cloud resources. These include versions of SW-G developed using DryadLINQ [9,10], Apache Hadoop (map-reduce) [9,10], and Twister (Iterative Map-Reduce) [8], each of which shows promising performance and scalability.

The pipeline model discussed here presents a flexible architecture which allows for the use of any of these Smith Waterman-Gotoh implementations, as well as other alignment and analysis tools as they become available. By exposing this entire pipeline framework through standard web service interfaces, the described portal system will enable the researcher to employ the appropriate computational strategy based on the specific problem at hand and available computational platforms, serving as a simple gateway to grid, cluster, and Cloud resources.

III. BIOSEQUENCE CLASSIFICATION WORKFLOW

The problem at hand is to classify biological sequences of either DNA or Proteins into clusters of similar nature. Eventually, this would help biologists to analyze various characteristics of these sequences in their domain of work.

A. *Classification Scheme*

Classification begins with the calculation of a value to indicate the similarity between each pair of sequences. This value is usually known as the distance between two sequences. Thus, a matrix of distances for each pair is generated and is fed to a clustering algorithm implementation. The clustering algorithm will assign a group number to each sequence such that sequences with similar distances to other sequences fall into the same group.

The previous steps are the essence of the classification process. A visualization step, however, dramatically increases the usefulness of such results for analytical purposes. Therefore, a point visualization based on the same distance matrix is attached to the chain of algorithms. The goal of the visualization algorithm is to find 3-dimensional (3D) coordinates for each sequence such that each pair of points has the same distance as the corresponding two sequences in the distance matrix. A visualization tool is then able to produce a 3D picture depicting the clustering of sequences.

The classification scheme described here is shown in Fig. 1.

B. *Software Pipeline*

Several pieces of software work in a flow (pipeline) to complete one classification task. Primarily, it contains MPI.NET-based algorithm implementations of pairwise sequence alignment (distance calculation), pairwise clustering, and multi-dimensional scaling. These algorithms are run on Microsoft HPC Clusters, which require manual labor to configure and submit classification jobs. We have implemented a job configuration and submission tool to overcome this effort. The tool provides a simple Graphical User Interface (GUI) to select the suite of algorithms and the computer cluster nodes. It also enables configuring the algorithm parameters via a simple interface. Once the user is happy with the configuration he/she can submit the job to the cluster from the tool itself. The job is saved to the local machine, which makes it easy to perform a re-run or to use some of the algorithm configurations for a new job. Once the job is completed, the user can retrieve results from the cluster (manually at present) and use the visualization tool, i.e. PlotViz, to interact with a 3D representation. Fig. 2 encapsulates the use of these software pieces in the big picture.

Figure 1. General Bioseqeunce Classification Scheme

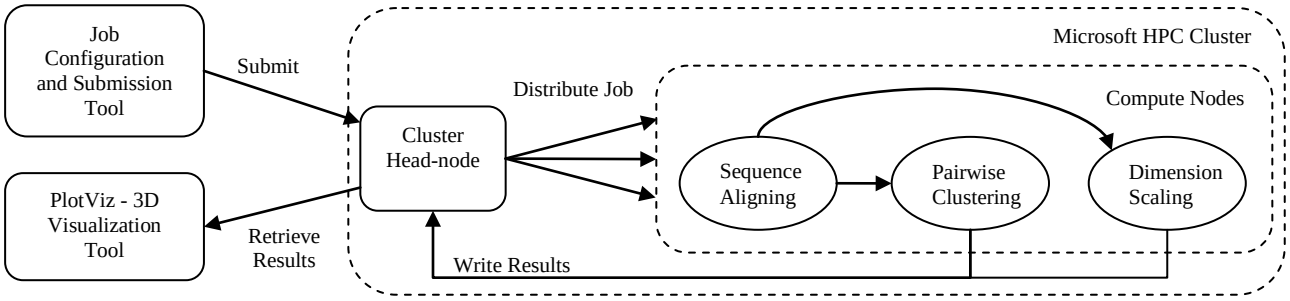

Figure 2. Software Collection in Bioseqeunce Classification

IV. Service-Oriented Architecture

Software utilities designed to chain multiple applications into a coherent workflow, such as the biosequence classification scheme described above, have the potential to dramatically reduce the manual effort required to perform sophisticated scientific analyses of large result sets. However, in order to successfully manage the deluge of data generated by modern laboratory techniques, scientists are becoming increasingly reliant on high performance computing (HPC) resources, such as Clouds and more traditional Grids. Managing data and computational jobs on distributed, and potentially heterogeneous, platforms can rob scientists of valuable research time as they wrestle with the manual effort of tracking their work.

To help overcome this bottleneck, the SALSA Portal is being developed as a suite of generic web services for creating and managing HPC jobs on a wide variety of computing platforms. Because these utilities are presented as services, they are readily consumable by many types of software clients [11], including web sites and desktop applications, allowing end users to compose the services into utilities that satisfy their individual needs. Furthermore, the service architecture is fully extensible, enabling access to new analysis tools and computing platforms, as they become available, with minimal new development effort.

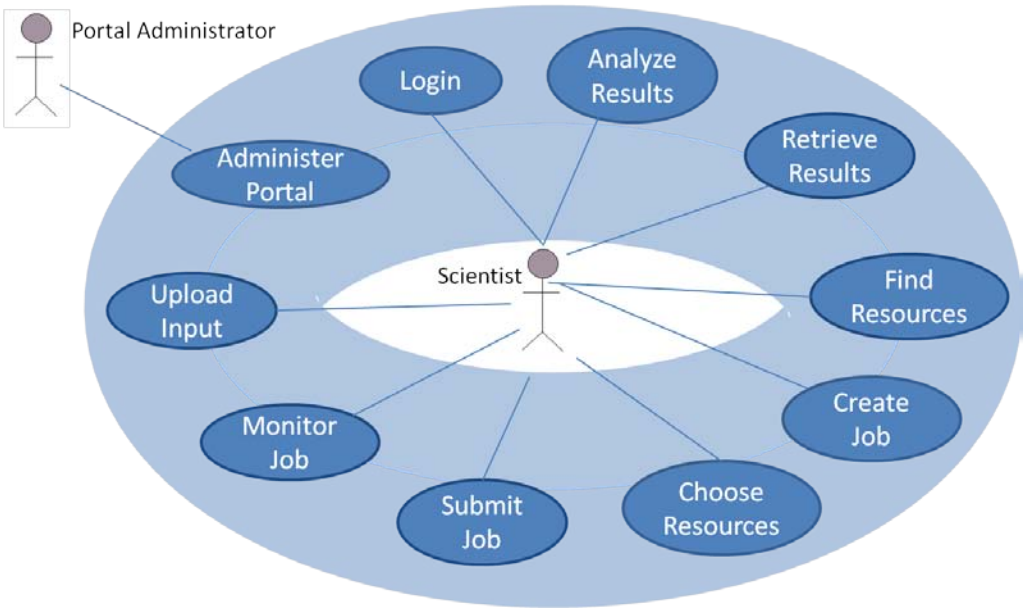

Figure 3. This use-case diagram shows the functionalities for high-performance computing resource and job management provided by the SALSA Portal web services Collection in Bioseqeunce Classification

A. Portal Functionality

In order to provide access to a wide array of scientific software tools and computing platforms, the Portal services must present flexible, extensible interfaces. Even so, certain foundational functionalities are required for managing computational jobs under most conditions, and these *use cases*, as depicted in Fig. 3, form the basis for the Portal's design. The major areas of functionality provided by Portal services are as follows:

- *Security:* Access to computing resources and data must be managed to ensure the integrity of the sytems and the privacy of important research data.
- *Resource Discovery:* The system needs to provide information to users about which computing platforms and software tools are available to them at any given time.
- *Data Transfer:* Users need the ability to transfer input data to the HPC machines where their jobs will be run and to retrieve results from these machines when jobs have completed.
- *Job Management:* In order to take full advantage of available resources, users need the ability to build, submit, monitor, and cancel computational jobs.
- *Portal Administration:* The system must provide an interface for administrative users to conduct basic portal maintenance activities, including the management of portal configurations, user accounts, and user jobs.

B. Portal Design and Architecture

Sustained rapid advances in the physical sciences necessitate the timely development of new software tools and continual improvement of existing utilities in order to allow scientist to analyze increasingly large results sets. Likewise, available computing resources are generally in a state of slow flux as existing clusters are expanded or reconfigured, new grid resources become available, and Cloud Computing options are explored. Because of this variable computing landscape, the Portal design must provide a general framework for managing HPC jobs, but also be readily extensible for future viability.

To address these requirements, the SALSA Portal was designed as a Service-Oriented Architecture (SOA) arranged in logical layers [12], as shown in Fig. 4, allowing for software reuse and the composition of available functionalities. Each layer consists of a set of software components that may be used to carry out the various actions required for job and resource management. In order to ensure the continued integrity of Portal operations, components in one layer may only access other components from its own layer or from the layers below it in the logical hierarchy. Each of the Portal layers is discussed here in turn.

- *Resource Layer:* This is a heterogeneous layer consisting of foundational resources accessed by other layers of the Portal, including HPC machines, Active Directory services (in a Windows environment), and various databases, either existing or created specifically for Portal use. Transparent access to these resources is the overriding aim of the entire Portal system.
- *Resource Access:* The Resource Access layer consists of a set of software components that serve as connectors from the higher-level layers in the system to the Resource Layer below. These components are generally designed to be services but may be simple libraries, depening on context and complexity.
- *Workflow Logic:* The software components in this layer implement the logic necessary to carry out the user-facing functionaities discussed in section III.A. In general, a *Manager* component orchestrates the implementation of a use case by invoking one or more *Engine* components, which carry out specific logic. Because the Managers are client-facing, each is implemented as a web service, as are most of the Engines, allowing for agile use case composition.
- *Client:* This layer is not a part of the actual Portal system, but represents the ultimate presentation of Portal capabilities to end-users. Because the Portal exposes its Manager components as web services, many different types of client applications can be built which use these components, depending on the specific problem domain and user needs.
- *Utilities:* The Portal utilities are a set of software components that provide functionality needed by several other Portal components, such as logging and string functions. The utility classes do not strictly define a layer, as they are are available to components from any other region of the architecture.

C. Portal Implementation

Because of the flexibility that they offer in system design and composition, web services have become very popular, as an extension of the generalized SOA described in [12], across a broad spectrum of application domains and project settings. Not surprisingly, this popularity has led to the evolution of many different frameworks for implementing web services. Among the more widely used of these paradigms are those based on Java and .NET technologies, such as Apache Axis and ASP.NET web services, respectively ([13] and [14]).

While each of these technologies presents certain benefits and drawbacks, the emergence of the Windows Communication Foundation (WCF) provides developers with a comprehensive set of features that make WCF very attractive for enterprise-quality software production in a variety of settings, as in [11]. In the context of the SALSA Portal, WCF offers service-level security and logging, the ability to use WCF components as plain classes locally or deploy them as services, streaming data transfer, and the relative stability and low maintenance of the Internet Information Services (IIS) application container [11].

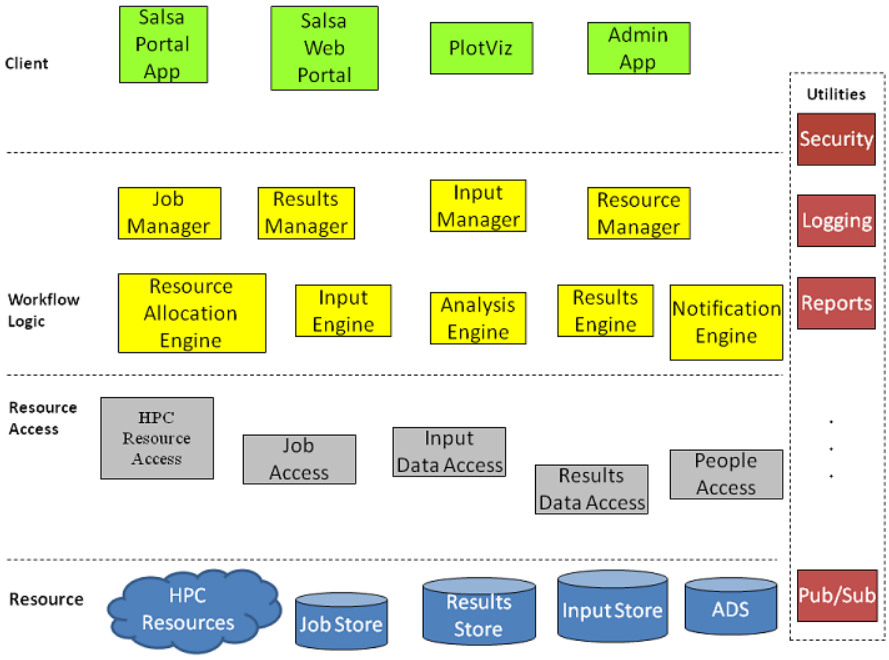

Figure 4. The multi-tiered, service-oriented architecture of the SALSA Portal services. All *Manager* components are exposed as web services and provide a loosely-coupled set of HPC functionalities that can be used to compose many different types of client applications.

Importantly, because the developer community for this project potentially includes undergraduate students or others with limited programming experience, WCF development also presents a low barrier to entry since it is tightly integrated with the familiar .NET Framework and the Visual Studio Iterative Development Environment (IDE). Consequently, all Portal web services have been developed using WCF.

V. Results

Extensive design work has been completed for both the biosequence classification workflow and the SOA Portal described here. Implementation is ongoing, but early versions of these software systems have yielded potential benefits in managing computational jobs for the biological sciences, as discussed below.

- *Biosequence Classification Workflow:* To this point, the application chaining described in section II has been implemented and deployed to various Windows HPC Server platforms. This pipeline allows researchers to perform Smith-Waterman-Gotoh (SWG) sequence alignment, followed by multi-dimensional scaling (MDS) and/or pairwise clustering (PWC) for a given set of sequence reads, all with only one set of input files and configurations. Without this construct, users would need to move input data to an appropriate location, configure and run SWG, potentially transfer SWG output files to a desired location, and then configure and run MDS or PWC for dimensional reduction. This pipeline reduces the necessary steps to the initial data transfer and the configuration of one job that controls the entire application stack.
- *SALSA Portal:* The initial implementation of the SALSA Portal, targeting local resources and the biosequence classification workflow described in this paper, has been completed. This version of the Portal consists of a suite of web services that make available to end users many of the basic functions necessary to manage high-performance computing jobs. As an example of client software which consumes these SALSA Portal web, a simple web site has been developed to present a basic job-control interface to the user. As shown in Fig. 5, this web client allows scientists to find available HPC resources and analysis tools, upload input files, submit and track jobs, and retrieve the results of the computational jobs from one simple interface.

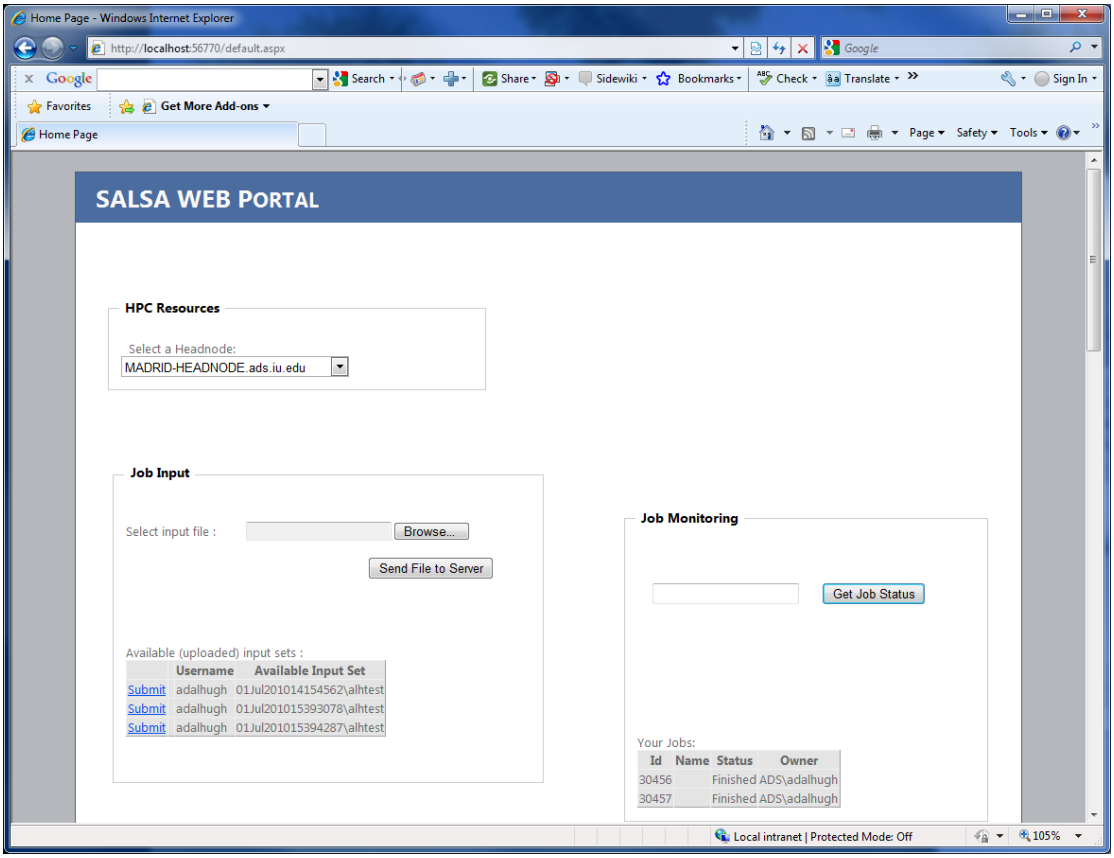

Figure 5. A simple web client that utilizes SALSA Portal web services to provide HPC resource discovery, file transfer, and computational job control functionalities to end users.

VI. Future Directions

Although architectures for the biosequence classification workflow and the SALSA web service Portal are now well-established, there remain many areas of improvement and advancement to be addressed in the near- and long-range future. Most of these efforts are focused on expanding system scope, reliability, and security in order to enable ready access to additional HPC resources such as Cloud platforms. These planned and ongoing activities are described here.

A. Portal enhancements

1) Streaming file transfer

The current version of the SALSA Portal services supports access to local HPC resources by in-house researchers, and, as such, utilizes a simple file transfer protocol for uploading input and downloading results. In order to make effective use of distributed computing resources, a more robust and ubiquitous file transfer solution is required. For this purpose, a streaming file transfer service based on WCF is being developed, which will allow for the upload and download of large datasets while limiting timeout errors typical when encountering file buffer size limits.

2) Expanded job creation interfaces

The initial version of the Portal job creation interface requires the user to pack his input files into a single ZIP file with a particular directory structure. Moving forward, maximum benefit can be gained from these services only if users are afforded the opportunity to build their jobs through a more intuitive interface that exposes the job settings of specific algorithms, as users are accustomed to manipulating for their local jobs. In addition, there is work to be done on supporting robust client layers for HPC resources other than the initial testbed which has been targeted here. The first thrust of this work will be to make the current desktop interface of the sequence classification workflow discussed in section I accessible through Portal services.

3) Access to common bioinformatics tools

As the Portal system becomes more robust, it will provide real value only by making resources available to actual researchers who need to use bioinformatics tools. As described here, the Portal is designed as a set of loosely-coupled services that allow such client applications to be

quickly developed. However, it will be helpful to future developers, as well as immediately impactful, to provide some fully-functional examples targeting specific high-use algorithms. In particular, a simple web interface allowing quick access to the BLAST algorithm against a number of standard sequence databases, running on a variety of HPC platforms, is planned for development in the near-term.

4) Security

Ensuring the security of system resources and scientific data is paramount to the integrity of a distributed software system. During the initial stages of SALSA Portal development, access has been handled via Windows Authentication through Indiana University's Active Directory system. As the system is expanded to target non-local platforms and to provide access for external users, a more expansive security model needs to be implemented. Various security mechanisms, including WCF options, are being explored.

B. Map-reduce pre-processing

In the current version of the sequence analysis pipeline presented here, the Message Passing Interface (MPI) implementation of the Smith-Waterman-Gotoh (SW-G) algorithm [6,7] is used to perform the sequence alignment step. While this version of SW-G is well-suited to share-memory machines and traditional clusters, it is not appropriate for use on distributed grid and Cloud platforms. To overcome this limitation, the pipeline will be generalized to allow on-the-fly selection of the Hadoop, Twister, and DryadLINQ implementations of SW-G, exposing the powerful combination of map-reduce and Cloud technologies for usage by biology researchers.

C. Service-based generic map-only Tasks

Many biosequence analysis pipelines rely on inherently data-parallel algorithms such as BLAST to enable genome annotation. Researchers can use these tools to take full advantage of HPC resources by distributing input datasets to available nodes and then running the same software on each node, simply concatenating results on job completion. In fact, this common map-only pattern is used in many existing workflows, but the effort involved in splitting input files and managing the related jobs typically requires tedious manual effort or a complicated scripting environment.

The Twister iterative map-reduce runtime [15] can be used for this type of problem to abstract the mapping of input to available computer resources. In certain cases, users may desire a specialized Twister implementation to address a particular problem that requires very specific analysis during the reduce step. However, in many cases, researchers simply need to run their data-parallel tools as efficiently as possible and then collect all of the results when the job has completed. The SALSA Portal will address this need by exposing a generic version of the Twister technology through a simple web service. This will allow researchers to run map-only jobs from any context in their workflow and take advantage of all available resources, whether clusters, grids, or Cloud platforms.

VII. CONCLUSION

The deluge of data produced by modern experimental techniques in the physical sciences is making it increasingly obvious that analysis and visualization no longer belongs solely in the realm of local laboratory computing resources. Instead, such problems must be approached utilizing state-of-the-art software tools running on distributed high-performance computing resources, both traditional clusters and various Cloud platforms. The substantial usage barrier for making full utilization of these resources often leaves researchers frustrated and with a mounting cache of raw data.

The combination of algorithm "chaining" to form analysis pipelines and web services to allow easier access to computational tools holds much promise for easing the current tedious data manipulation burden facing many researchers in the biology and health science fields. The extensible architecture of the SALSA Portal described here will allow for relatively simple expansion to support new software tools and computing platforms as they are developed in the coming years. This flexibility is vital to keeping pace with the proliferation of biological and health data and allowing researchers to focus their efforts on the scientific problems at hand rather than on the manual effort required to manage data movement and computational job submissions.

ACKNOWLEDGMENT

This work is supported by the National Institutes of Health under grant 5 RC2 HG005806-02.

REFERENCES

[1] M. Marguilies et al., "Genome sequencing in microfabricated high-density picolitre reactors," *Nature*, vol. 441, no. 7089, . pp. 376-380, Sep. 2005.

[2] B. Richter and D. Sexton, "Managing and Analyzing Next-Generation Sequence Data," *PLoS Computational Biology*, vol. 5 no. 6, Jun. 2009.

[3] M. Pierce, C. Youn, and G. Fox, "The gateway computational web portal," *Journal of Concurrency and Computation: Practice and Experience*, vol. 14, no. 13-15, pp. 1411-1426, Nov.-Dec. 2002.

[4] J. Orvis, et al., "A web interface and scalable software system for bioinformatics workflows," *Bioinformatics*, vol. 26, no. 12, pp, 1488-1492, Jun. 2010.

[5] O. Kaiser, et al., "Whole genome shotgun sequencing guided by bioinformatics pipelines – an optimized approach for an established

technique," *Journal of Biotechnology*, vol. 106, no. 2-3, pp. 121-133, Dec. 2003.

[6] T. Smith and M. Waterman, "Identification of common molecular substances," *Journal of Molecular Biology*, vol. 147, pp. 195-197, 1981.

[7] O. Gotoh, "An improved algorithm for matching biological sequences," *Journal of Molecular Biology*, vol. 162, pp. 705-708, 1982.

[8] J. Ekanayake, T. Gunarathne, X. Qiu, "Cloud Technologies for Bioinformatics Applications," accepted to the *Journal of IEEE Transactions on Parallel and Distributed Systems*, Special Issues on Many-Task Computing, accepted for publication.

[9] J. Qiu, et al., "Data Intensive Computing for Bioinformatics," in *Data Intensive Distributed Computing*. Hershey, PA:IGI Global: 2011.

[10] G. Fox, et al., "Biomedical Case Studies in Data Intensive Computing," keynote talk, *1st International Conference on Cloud Computing (CloudCom 2009)*, Beijing Jiaotong University, China Dec. 2009.

[11] J. Lowy, *Programming WCF Services*, 1st ed. Sebastopol, CA: O'Reilly Media, 2007.

[12] T. Erl, *Service-Oriented Architecture (SOA): Concepts, Technology, and Design*, 9th ed. Crawfordsville, IN: R.R. Donnelley, 2009.

[13] K. Tong, *Developing Web Services with Apache CXF and Axis2*, 3rd ed. TipTec Development, 2010.

[14] A. Ferarra and M. MacDonald, *Programming .NET Web Services*, 1st ed. Sebastopol, CA: O'Reilly Media, 2002.

[15] J. Ekanayake, et al., "Twister: a runtime for iterative mapreduce," Proceedings of the First International Workshop on MapReduce and its Applications of ACM HPDC 2010, June 20-25, 2010, Chicago, Illinois.