Published in final edited form as:
Proceedings (IEEE Int Conf Bioinformatics Biomed). 2009 November 1; 1-4(Nov 2009): 392–395. doi: 10.1109/BIBM.2009.84

Enabling Data Analysis on High-Throughput Data in Large Data Depository Using Web-Based Analysis Platform – A Case Study on Integrating QUEST with GenePattern in Epigenetics Research

Terry Camerlengo, **Hatice Gulcin Ozer**, **Pearlly Yan**, **Jeffrey Parvin**, **Tim Huang**, and **Kun Huang**
The Ohio State University Columbus, Ohio, USA

Francisco Perez
The Wistar Institute Philadelphia, Pennsylvania, USA

Mingxiang Teng, **Lang Li**, and **Yunlong Liu**
Indiana University Indianapolis, Indiana, USA

Tahsin Kurc
Emory University Atlanta, Georgia, USA

Abstract

Enabling data analysis in large data depositories for high throughput experimental data such as gene microarrays and ChIP-seq is challenging. In this paper, we discuss three methods for integrating QUEST, a data depository for epigenetic experiments, with a web-based data analysis platform GenePattern. These methods are universal and can serve as an exemplary implementation resolving the dilemma facing many similar database systems in integrating data analysis tools.

Keywords

high-throughput database; GenePattern; ChIP-seq

I. Introduction

During the past decade, advancement in high throughput experimental technologies such as gene expression microarray and massive parallel sequencing has revolutionized biomedical research as it allows the investigator to carry out genome wide study on key biological processes in a single experiment at a relatively cost. However, these technologies also raise new challenges to biomedical informatics in two aspects: data management and data analysis. While a microarray experiment can generate megabytes of the data, the output of the massive parallel sequencing experiments is in the scale of tens of gigabytes while the raw data is usually in the size of 1-2 terabytes. Therefore for data management, the focus is to develop efficient database systems to allow the user to query for datasets from multiple platforms, experiments as well as values associated with individual genes or gene groups. A well-known example of such data depositories include the Gene Expression Omnibus (GEO) maintained by NCBI. In addition, there are numerous local and legacy data depositories that were implemented by individual research groups and institutions.

While the databases are convenient for users to store and retrieve the data, it is not always easy for the user to carry out analysis especially given that the development of data analysis is a highly dynamic process. it is not realistic or feasible to define a static data analysis

model that can be integrated into the database. As a result, most databases can only provide a minimal set of analysis tools such as the simple Student t-test function in GEO.

For advanced analyses, for most of the time, the user has to download the data and the software or computer codes for the analysis, carry out the analysis, and store the results on a local computer. This is not only clumsy and time-consuming, it also causes problems such as poor documentation on the analysis protocol, different choice of parameters, lack of repeatability of the analysis results and sometimes loss of results. In addition, the analysis algorithms are often developed in different languages such as R, Matlab, Python, Perl, and Java, which is usually a big hurdle for regular biomedical researchers to properly install and utilize these tools.

One effort to resolve this issue is the web-based platform GenePattern developed by the Broad Institute [1]. It allows codes for data analysis algorithms in different programming languages (e.g., Java, R, and Matlab) to be uploaded as standard modules. Different modules can be organized into a data analysis pipeline and saved in GenePattern. A user can enter GenePattern, upload the data, and apply the analysis modules or pipelines without the need to support different languages. This not only enables easy sharing of the analysis modules, it also enforces standard interface between modules and improves reproducibility of the analysis results.

So an important question is, “Can we integrate existing, legacy databases to GenePattern so that the users of these databases can carry out data analysis much more smoothly?” Even more important, the analyses would be standardized and the results can also be stored in the databases.

QUEST (http://bisr.osumc.edu/) is developed at the Ohio State University as a data management and ad-hoc query system that catalogues and stores microarray and massive parallel sequencing data from platforms. The QUEST project started out as a data-sharing portal for the NCI ICBC Center between Indiana University (IU) and The Ohio State University (OSU) for epigenetics study. It is deployed as a central data portal for different shared resources such as the Illumina sequencing facility at the OSU Comprehensive Cancer Center. One characteristics of QUEST is that it enables users to build complex queries using an intuitive graphical user interface (GUI) without the need to write tedious SQL statements. As researchers acquire more data of differing types, they can add it to their “data stores” and QUEST will reflect the new entities in its GUI, allowing users to query the newly acquired data types without a new programming endeavor. However, a missing functionality in QUEST is data analysis.

Our goal was to leverage the existing analytical platform in GenePattern with the data management functionality of QUEST to enable two-way communication between QUEST and GenePattern for our users. First, a regular investigator can login to QUEST, select the datasets to be analyzed, invoke the analysis workflow in GenePattern and get the results back in QUEST. Second, an advanced user (e.g., bioinformatician) can enter GenePattern, obtain data from QUEST, carry out the analysis, and store the results locally in GenePattern. This will also allow fast test and parameter tuning of new algorithms. In order to achieve these goals, we identified three modes for QUEST and GenePattern integration (Figure 1) and implemented them. Even though our work is developed for the specific database system QUEST, the methods presented in this paper are universal and can serve as exemplary implementation for similar database systems to solve the data analysis issue.

II. Methods

In this section, we describe the implementations of three distinct modes for QUEST and GenePattern integration:

- invoking data analysis in QUEST and transferring the data to GenePattern using a URL;
- invoking data analysis in QUEST and transferring results from GenePattern back to QUEST (RPC style);
- invoking analysis in GenePattern and directly transferring data to GenePattern (XML-RPC).

The XML-RPC mode requires a GenePattern module (i.e. Quest Importer) to initiate a request on an analysis workflow from QUEST to GenePattern. The last two modalities involve passing data from QUEST to GenePattern. The RPC-style invocation can exhibit two way communications, where QUEST invokes a GenePattern module and receives a GenePattern response, which can be further processed. These techniques are described below.

Mode 1: invoke data analysis in QUEST and transfer the data to GenePattern using a URL

In this scenario, data is requested in QUEST and bundled into an archive (i.e. a zip file) and published to a GenePattern accessible folder via URL. QUEST supports archives generated from two data sources: raw data files (e.g., Figure 2) and query results (e.g., Figure 3). Typically data requests consists of raw data files but can also be query results aggregated into a file format and placed into an archive. Once an archive is generated in QUEST, a link is constructed in a format so that both the archive URL and analytical module are indicated. The user can select the GenePattern module of interest in QUEST by accessing a drop down list with registered modules (e.g., Figure 4). The GenePattern user can then tweak the parameters and run the module or pipeline. There is no further communication between GenePattern and QUEST from this point on for this integration technique.

Mode 2: invoking data analysis in QUEST and transferring results from GenePattern to QUEST (RPC style)

QUEST is programmed in ASP.NET/C#, whereas GenePattern is programmed in Java. To bridge the technical barrier between the CLR and the JVM, Quest makes use of the Mono project (http://www.mono-project.com/Java).

The Mono project provides facilities for translating and compiling the GenePattern server *.jar* files into a GenePattern server *.dll* files (i.e. A Dynamic Link Library). A *.dll* can be referenced in an ASP.NET project as a managed code assembly and remote calls can be made directly from QUEST. Once the *.dll* is created we can reference classes from the client package by including:

using org.genepattern.client;

Figure 5 illustrates how QUEST invokes a GenePattern module programmatically by creating a local GenePattern client and passing a series of parameters, including module name and archive location. This process uses the local *jobResult* attribute to store return value information. This attribute enables user to subclass the class *GenePatternTask* to handle specific return values. Otherwise, GenePattern task can handle any one-way communication request where the return value is not important.

Figure 6 is a code-snippet that demonstrates how to handle the return value from a GenePattern remote procedure call via sub-classing. By developing a base class called *GenePatternTask* in QUEST, we can abstract the common underlying behavior required in QUEST-to-GenePattern interaction. By extending the base class, we can add support for return values from specific modules. In this example, the subclass *SolexaDownloadGenepatternTask* simply calls the base class *Do( )* function and accesses the job result parameters for further processing.

Mode 3: invoking analysis in GenePattern and directly transferring data to GenePattern via XML-RPC

In this mode, Genepattern makes data requests to QUEST thru an XML-RPC interface (http://www.xmlrpc.com/). XML-RPC is a technology based on remote procedure calls, which encode requests as XML over an HTTP transport. XML-RPC is a simple and low-overhead protocol to initiate distributed communication between disparate systems.

The Quest Importer (Figure 7), allows GenePattern users to export ChIP-Seq data from QUEST into GenePattern. It requires the following inputs: username, password, sample Id, and sequencing File Type (e.g., realign, seq, prb). The Importer will invoke the XML-RPC interface to QUEST *SolexaChipSeqRPC.ashx* (http://bisr.osumc.edu/QUEST/Public/SolexaChipSeqRPC.ashx). Once QUEST receives the request then the file types for the sample id specified will be archived and published to a URL that GenePattern can access. The data is imported into GenePattern and is available for further analysis.

III. Results

The QUEST-GenePattern integration is used in epigenetics study especially in analyzing large set of ChIP-chip and ChIP-seq data generated by Solexa sequencers. The ChIP (chromatin immunoprecipitation) experiments studies protein-DNA interaction over the entire genome. Currently our proteins of interests include estrogen receptor, RNA polymerase II (Pol II), and histone makers with different methylation status (e.g, H3K4me2, H3K27me3).

As we discussed previously, in the Mode 1, we can carry out various data analysis by invoking analysis tools in GenePattern. Figure 8 Top shows an example for visualizing ChIP-chip data in QUEST. The intenities for each probe in the ChIP-chip experiment across ten samples can be visualized using the tool in GenePattern. In addition, a user can zoom into any specific region in the genome using the user defined query.

Currently we use the Mode 2 to set up automatically data downloading from outside sequencing centers. Once we receive a notice from the sequencing center about the availability of a set of ChIP-seq data, we logon to QUEST and invoke a data downloader implemented in GenePattern. The downloader then retrieves the data from the sequencing center FTP server to the GenePattern server, send a notice to QUEST, and then automatically transfer the data to QUEST.

Finally for Mode 3, we have implemented the ChIP-seq data annotation pipeline xIP-Seq [2,3] in a local GenePattern server, using the QUEST-importer, the researchers can request data from QUEST (Figure 6) to perform data analysis (Figure 8). In the bottom of Figure 8, two sets of ChIP-seq data for the Pol II binding on two different breast cancer cell samples were extracted. The left panel in the bottom of Figure 8 indicates the parameters used to set up the GenePattern pipeline. The right panels shows the results on the tag counts over all the

genes for Pol II bound segments and the genes with significant differential binding quantities (in red dots) obtained using a mixture Poisson model fitting algorithm [2].

IV. Conclusion and Discussion

In this paper, we discussed three methods for integrating QUEST, a data depository for high throughput experiments, with online data analysis platform GenePattern. These methods are universal and can serve as exemplary implementation resolving the dilemma facing many similar database systems in integrating data analysis tools. They can be particularly useful for managing and analyzing the new massive parallel sequencing data. Currently we plan to expand the QUEST system into a unified caGrid framework to facilitate grid-enabled computing.

Acknowledgments

This work is partially supported by NCI ICBP grant (U54CA113001) and the PhRMA Foundation Research Starter Grant in Informatics.

References

[1]. Reich M, et al. GenePattern 2.0. Nat. Genet 2006;Vol. 38(5):500–501. [PubMed: 16642009]

[2]. Feng W. A Poisson mixture model to identify changes in RNA polymerase II binding quantity using high-throughput sequencing technology. BMC Genomics 2008;vol. 9(Suppl 2):S23. [PubMed: 18831789]

[3]. Wang, X., et al. xIP-seq platform: an integrative framework for high-throughput sequencing data analysis. Proc. Ohio Collaborative Conference on Bioinformatics (OCCBIO); IEEE Press; 2009.

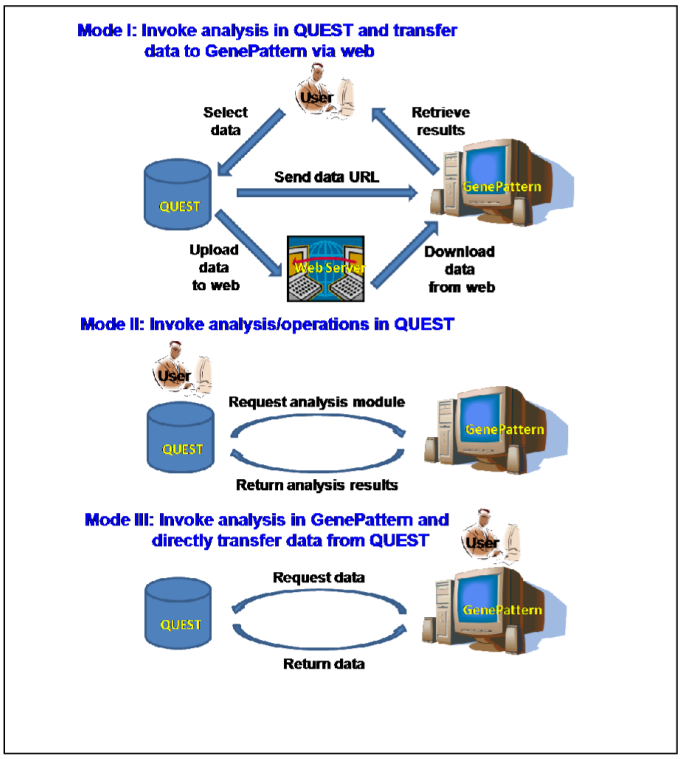

Figure 1.
The three modes of integration for QUEST and GenePattern.

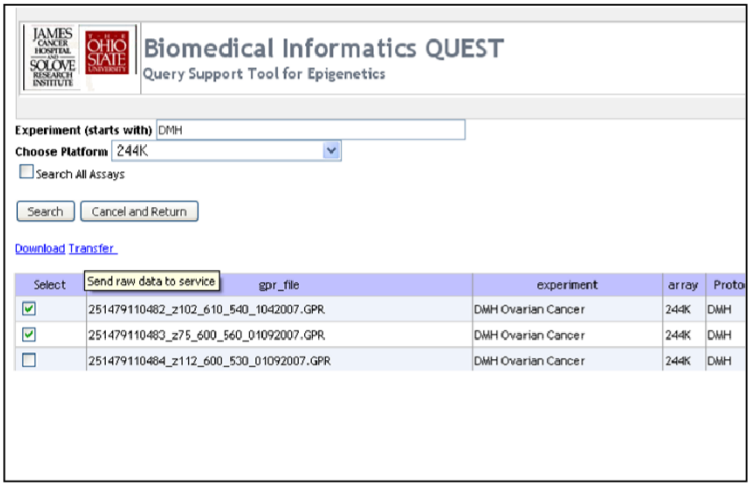

Figure 2.
Selecting raw data files in QUEST for export to GenePattern.

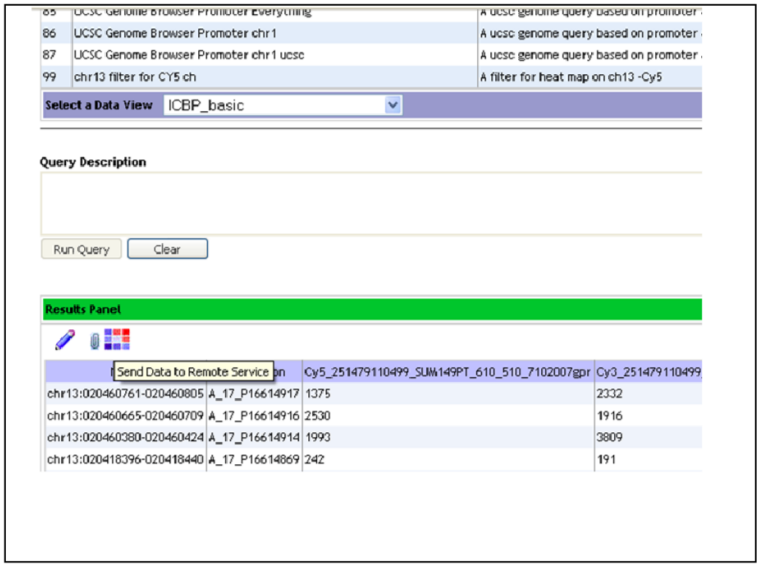

Figure 3.
Aggregating results from a query for exporting to GenePattern.

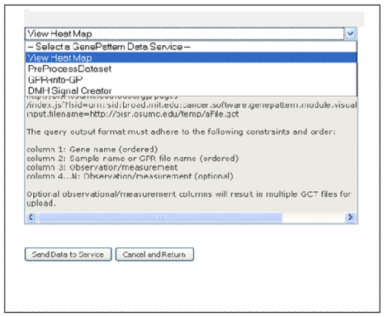

Figure 4.
Selecting the GenePattern destination module (where the data is to be sent).

```
public override void Do()
{
    bool writeOK = false;

   // Begin GenePattern invocation.
   try
   {
     //invoke  module class
     GPClient gpClient = new GPClient(this._server,
      this._UserName, this._password);
     JobResult result =
     gpClient.runAnalysis(this._GenePatternModule,
     this._Params);

     //save to object attributes
     this._JobId = result.getJobNumber();
     this._jobResult = result; //save
   }
   catch
   (org.genepattern.webservice.WebService
   Exceptionwexc)
   {
     Log(wexc.Message);  //log something...
   }
   catch (Exception ex)
   {
     Log(wexc.Message);  //log something...
   }
}
```

Figure 5.
Base Class *Do( )* function for invoking a GenePattern request from QUEST.

```
public class SolexaDownloadGenepatternTask :
 GenePatternTask
public override void Do(){
      base.Do();
       int jobId = _jobResult.getJobNumber();
       if (jobId > 0)
       {
           java.io.File[] files =
            _jobResult.downloadFiles(jobId.ToString());

           string[] strFiles =
           System.IO.Directory.GetFiles(jobId.ToString());

            // Copy the files and overwrite destination files
           // if they already exist.
            foreach (string s in strFiles)
           {
               //do something
           }
       }
}
```

Figure 6.
An example demonstrates the use of subclassing in QUEST-to-GenePattern interaction.

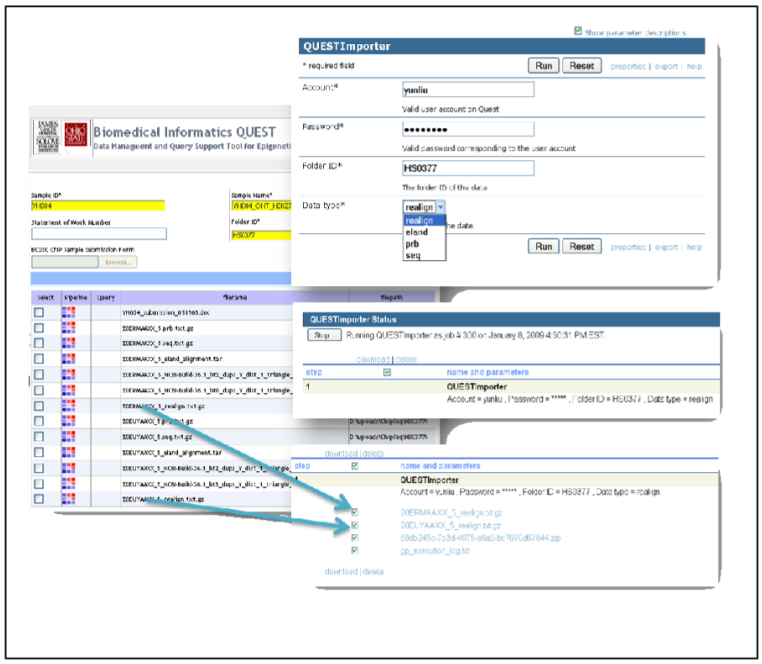

Figure 7.
The QUEST Importer in GenePattern.

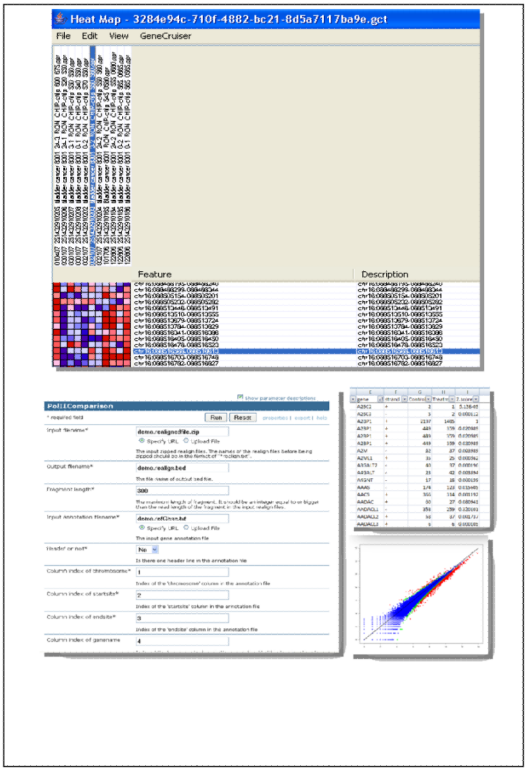

Figure 8.
Top: Heatmap for a ChIP-chip dataset in QUEST generated in GenePattern. Bottom: Examples of data analysis results for ChIP-seq data.