A Science Gateway - MOWIC: Modern Web Interface for Cluster

Daniel Cleland
Department of Computer Science
University of Kentucky
Lexington, KY 40503
1-859-699-2796

Chi Shen
Department of Computer Science
Kentucky State University
Frankfort, KY 40601
1-502-597-6083

ABSTRACT

High performance computing has been developed over the years providing researchers with the most powerful computational machines available at a given time. The National Science Foundation, which directs research and development of these resources, has outlined goals for developing sustainable and extensible HPC systems and services, [1][2]. These goals include the developments of systems that support individuals as well as groups to more easily use computation [1]. Both the user interfaces and the detail of building custom software prove to be a hurdle for researchers who don’t have access to computational human resources. Making these things easier to use for non-computer experts would increase the usage of existing resources and software. Making completed programs accessible via the web saves resources by encouraging researchers to reuse existing code. In this paper, we developed a science gateway --- MOWIC, the Modern Web-based Interface for Clusters, which attempts to meet many demands for research on parallel systems. It requires only that the user has a web browser. It can easily used with most hardware and software. MOWIC tool attempts to prototype an interface that can tie together all computational programs and visualization tools in research clusters.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *client/server and Distributed applications*

Keywords

Science gateway, User interface, Web design, Parallel program, Cluster.

1. INTRODUCTION

Cyber infrastructure is expanding and improving. The clusters that fulfill the needs of today’s computational research get larger and more sophisticated. Outreach programs work to promote awareness and provide training to increase the number of people utilizing the current infrastructure [3]. The interface for much of this power remains difficult and requires expertise in Linux shells, parallel programming or computational software packages. Preparing data and arranging it in such a way that makes it usable for publication or presentation can be tedious. Science gateways allow users to connect to datasets and HPC code but are much localized to the particular site. Being able to access information from existing datasets, with existing programs, in a common interface reduces costs and improves efficiency. This also allows for recreation of research providing for more transparency and enabling verification. A modern science gateway should incorporate features of modern web sites, tools to view and visualize data, user authentication and connections to run the programs needed, be accessible from anywhere and be user friendly.

MOWIC sets out to investigate the feasibility and possibility of creating a web based user interface to a high performance cluster can be made and provide the functionality needed by researchers to improve the human computer interaction with that cluster. This functionality includes running parallel codes, organizing the resulting datasets and allowing visualization of those datasets. This science gateway should focus on providing the end user an improved experience and productivity over existing command line interfaces.

The next section will introduce some backgrounds of other approaches for Science Gateways. Section 3 will outline the steps of system design. Experimental examples will be given in Section 4. Some issues related to the usage of MOWIC will be discussed in Section 5. Finally, future work for the next step of MOWIC development is presented in Section 6.

2. BACKGROUND

UNICORE, the Uniform Interface to Computing Recourse outlined primary goals for a universal interface to various HPC systems was designed in order to allow German supercomputer center users to access their clusters in a more user-friendly manner. UNICORE developers determined that such an interface would need to hide the system details, secure the connections to the clusters, be usable by scientists without them knowing the details of a particular system, support the batch schedulers and shells at the partner sites, and not require computing centers to dramatically change in order to work with the interface [4]. The issue with UNICORE still lies in its requirement of a downloaded java application. Java is a wise choice as it is such a portable language; however it is still a downloaded application that must be setup by each user. New features, bug fixes and versions are not updated until the user installs the new application.

SimpleGrid is a science gateway designed to be a reusable interface between web portals and applications. It allows for the handling of credentials, running an application, transferring a file and visualizing a dataset with JFreeChart and Google Earth/Map base visualizations. Work at the time of publication was being done to incorporate large scale ParaView visualization as well [5]. With goals set of encouraging and supporting the development of sciences gateways, a focus should be set more on supporting non-computer scientists use and navigate the tools, applications and information in a method that requires as little computer expertise as possible.

HUBzero is a project out of Purdue University which aims to provide social features, applications and tools through a Science Gateway. It uses open source software such as Apache, MySQL, PHP, and Joomla. HUBzero aims to bring a social aspect to the science gateway as well, providing features for collaborations between researchers, students, faculty and the general public. The focus of HUBzero is supporting collaboration between users, a friendly and powerful content management interface as well as visualization tools and simulation development with the rapture toolkit. The software allows for users to easily use and navigate tools to visualize their simulations [6]. HUBzero is greatly visually appealing and perhaps the best science gateway package available currently. It focuses heavily on collaboration and ease of use. The help and support features are excellent as well as the aesthetics of the Joomla based interface. Ratings, blogs, comments and wikis help dramatically for web enthusiast scientists to share their information and receive feedback. The Rapture toolkit that is used to visualize data is excellent and allows for easy integration of visualization tools to the gateway. This feature is critical the end user a strong catalyst to its success. More concentration should be places at enhancing the overall experience of the scientist which should include logging in to run jobs, organizing those data sources and then visualizing. In a social media world, enabling the embedding of those visualization tools into another location would also be a much desired feature.

Superbroker is a Netherlands based web interface for clusters that allows for many useful features. It is designed to make a single interface that is easy to user, to work for many different locations while providing the same experience to the user. The browser communicates with the server side part of the interface, which in turn passes parameters and runs the HPC code [7]. Superbroker was created with the concept in mind that "remote computing resources should be accessible without any technical barrier: without the need to install extra software and accessible through a simple graphical user interface. This structure of designing for non-computer savvy users, allowing the execution of applications, tools, and organizing files and separating client and server design is critical for a science gateway. It allows interface, web, visualization, and scientific developers to all work very independently. The Gateway Interface system should adapt to the user as well as the cluster, acting as an interpreter between the two.

MOWIC further develops the design of Superbroker. MOWIC allows for ready-to-run applications to be used remotely through a web interface which handles file management just as Superbroker. Beyond that, MOWIC allows for installation and integration of visualization tools in the same simplified interface as the numerical computation applications that it uses. MOWIC is also designed to be portable and easy to install and modify. MOWIC lays the foundation for powerful internet enabled collaboration through the interface. Rather than be an interface to a scientific application, MOWIC is an interface to a high performance cluster.

	Unicore	Simple Grid	HUB zero	Supr broker	Mowic
Target encapsulation	**Grid**	**Grid**	**Site**	**Grid**	**Site / Grid / Cloud**
Platform	**Java**	**Web (php, js, html)**	**Web (Joomla CMS)**	**Web (Java Applets)**	**Web (php, js, html)**
Run HPC jobs	**Yes**	**Yes**		**Yes**	**Yes**
Instant updates		**Yes**	**Yes**	**Yes**	**Yes**
Inherent Mobile Support		**Yes**	**Yes**		**Yes**
Extensible design			**Yes**	**Yes**	**Yes**
Shareable tools (modules)			**Yes**	**Yes**	**Yes**

Table 1 Comparison with other proposed and existing systems

In comparing existing systems and designing the MOWIC prototype many features, center around user experience, were analyzed. Such a system should be usable to power a single site, an HPC grid or to build a cloud-like HPC interface. Existing solutions like Unicore and SimpleGrid seem to focus on accessing HPCs across a grid to run applications. HUBzero provided many advanced features and extensibility without direct access to the high performance applications. Superbroker provided excellent access to whatever and HPC researcher would need though it was a central system that was not shareable to be used on a single site and used some dated web technologies, being over a decade old.

3. DESIGNING THE SYSTEM

3.1 Requirements

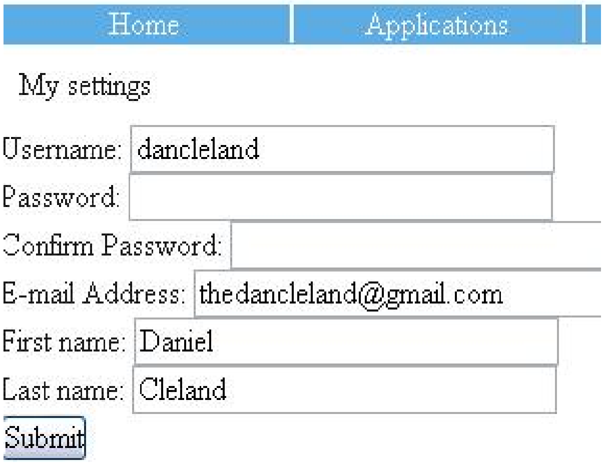
Hardware requirements for MOWIC are designed to be light. It requires a web server, of which Apache was used in both CentOS as well as in Ubuntu. There is no clearly definable minimum hardware requirement for the server to host the interface. The processing is all text based and web calls are asynchronous and may run in the background. It is recommended that at least a 500 MHz computer with 128M of RAM be used as the web server. The web server must have installed MySQL [8] and PHP [9], both of which are freely available for download at www.mysql.com and www.php.net respectively. It is expected that most existing Linux servers will suffice.

HTML and CSS provide the basic visual aesthetics of the interface. These presentation document formats are easy to learn, develop and maintain. The W3C maintains the standards for these, most of which can be seen at the W3 schools online learning site [10]. JavaScript is used to control the application communications to the server. This is done with a custom library file and jQuery. PHP provides server side control for the system. It processes requests from the browsers and sends information on to the cluster or to the MySQL database. Information sensitive to the system is processed only in PHP so that it is never passed to the user.

The server gains access to the environment through a web user with a home directory. This user is setup as a normal cluster account, except the account is setup as a group member of the web server. This give the web server access to the data files and execution privileges to the programs found in the user directory. This will allow the site to have to make minimal change in the way they function. As long as that user can access MPI, batch schedulers, etc, then the web users will be able to as well. The system will sort out what files and programs each user has access to in that directory.

3.2 Structure and Design

MOWIC was developed and tested on the small commodity cluster at Kentucky State University. This cluster is made up of Dell workstation computers. It can be accessed at http://cscluster.kysu.edu/MOWIC. This is a Rocks cluster using CentOS, Apache, PHP, and MySQL. Rocks Cluster Distribution is an open-source software collection, including a Linux distribution - CentOS, that allows for easy creation of computational clusters. It is an NSF supported creation [11].



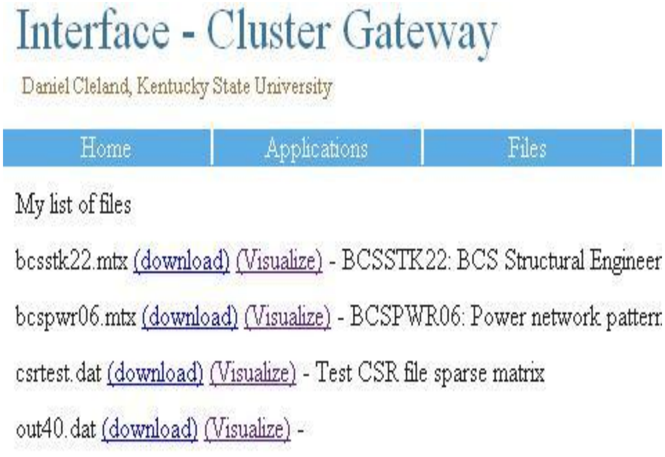
My settings

Username: dancleland
Password:
Confirm Password:
E-mail Address: thedancleland@gmail.com
First name: Daniel
Last name: Cleland

Figure 1 User setting screen

The menu is displayed in Figure 1 to reflect the simple nature of the system. It displays only five options which bring up the screen required, which are all downloaded to the client machine at the initial login of the system. A text banner is at the top of the screen with a welcome statement where the initial login box was upon entering the gateway. The Help screen displays information related to the system, the My Settings page provides the user a method to change settings and the Home page provides basic news and information.

The Files screen in Figure 2 displays the list of files that the user has access to. These files are listed with the filename and description. The options to download the files or visualize them with a tool are also populated. Selecting "(Visualize)" results a screen in Figure 3 where the user may select what visualization tool to use on the file. These are based on the known file format of the selected file.



Interface - Cluster Gateway

Daniel Cleland, Kentucky State University

My list of files

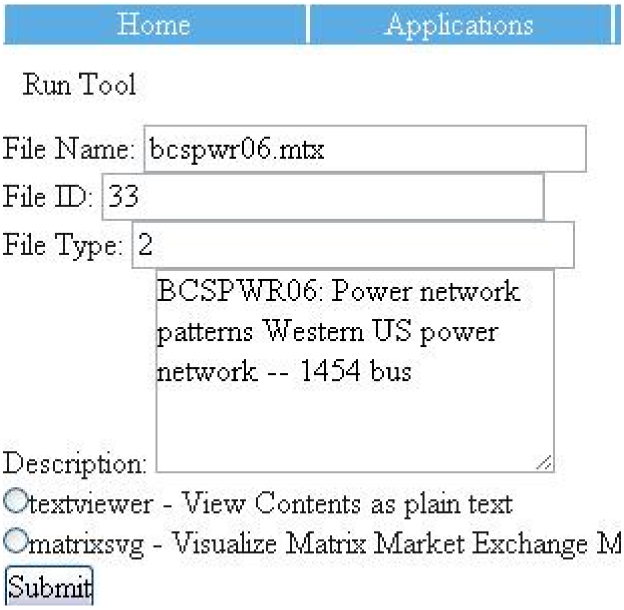
bcsstk22.mtx (download) (Visualize) - BCSSTK22: BCS Structural Engineer

bcspwr06.mtx (download) (Visualize) - BCSPWR06: Power network pattern

csrtest.dat (download) (Visualize) - Test CSR file sparse matrix

out40.dat (download) (Visualize) -

Figure 2 File screen provides users access to download or visualize the datasets that they have access to on the HPC



Run Tool

File Name: bcspwr06.mtx
File ID: 33
File Type: 2
Description: BCSPWR06: Power network patterns Western US power network -- 1454 bus

○textviewer - View Contents as plain text
○matrixsvg - Visualize Matrix Market Exchange M

Figure 3 Visualize screen allows users to select an installed visualization tool to view a data set with

If a user selects to visualize a data file a screen is loaded and populated with that file information. A bullet list is generated of tools that can be used to visualize that type of file. A general text view tool has access to view any file type. As for most parts of this system, the server is not needed to access this page; it is built into the downloaded application. All such pages of the gateway act by only querying the server when absolutely necessary.

Visualization tools are created to their specific task. An initial page of index.php is called in the tool folder. The tool is passed a POST variable named ‘filename’ that holds the name of the file that the user has requested to visualize. An ‘apps’ folder in the MOWIC directory holds all the visualization tools. These tools must have a unique name and be loaded in to the database as an installed tool. These are the only conditions of which are then free to implement their own functions, links, and features.

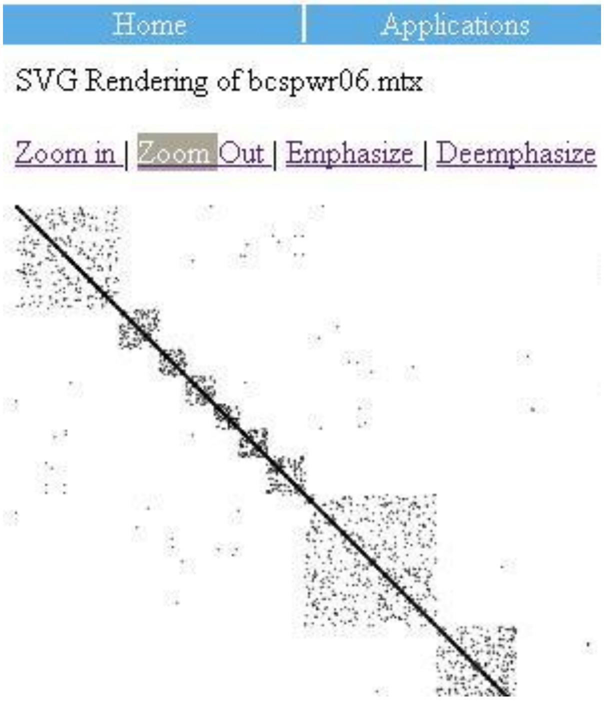

Home
Applications
SVG Rendering of bcspwr06.mtx
Zoom in | Zoom Out | Emphasize | Deemphasize

Figure 4 A visualization tool renders a sparse matrix in SVG format, displays it zoomed out to discover the patterns

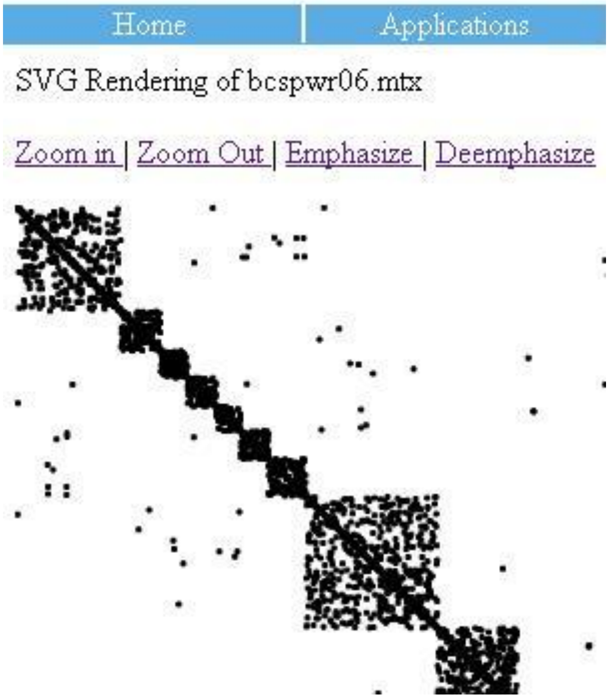
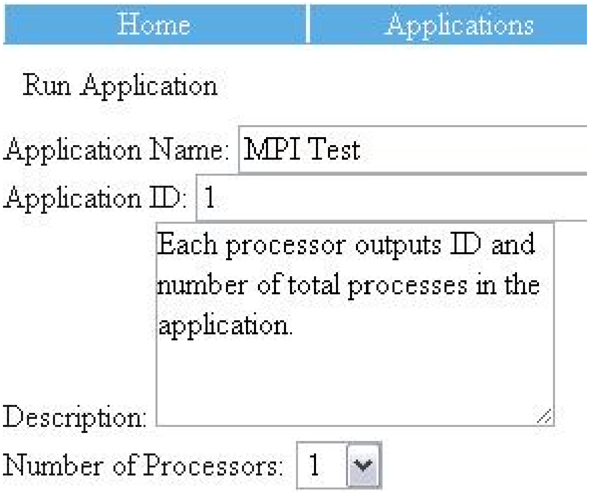

Home
Applications
SVG Rendering of bcspwr06.mtx
Zoom in | Zoom Out | Emphasize | Deemphasize

Figure 5 Visualization tool from Figure 4 displays the 'Emphasize' feature to increase the size of each data point

A sample tool was developed as a sparse matrix viewer for matrices in MTX matrix market format [12]. This tool provides the ability to zoom in and out of the visualization. The size of the points can be expanded or contracted with the 'emphasize' and 'deemphasize' links. All of this interaction is done in browser without querying the server. This is done through the nature of SVG files as XML documents that can be modified easily with jQuery JavaScript. As such this relies on the processing power of the client computer and may not be suitable for extremely large datasets. The matrix is a 1454 by 1454 sparse matrix with 3377 non-zero entries. On an Intel Core Solo 1.86Ghz laptop with 1GB RAM the emphasize functions take approximately three seconds to update while on an Intel Core 2 Duo 2Ghz laptop with 4GB RAM the functions are completed in less than one second. The total code needed to develop this particular tool was 15 lines of php/html in *index.php* file, 57 lines of jQuery/JavaScript in *scripts.js*, and 52 lines of php/xml/svg in *svggenerator.php*. Some library MOWIC system functions were accessed by linking to the *global.inc.php* file included with MOWIC. This is an example of a visualization tool that is reusable for any matrix in that particular format that was developed in one afternoon. This could be easily modified or adapted for different formats. By changing roughly 25 lines of code in *svggenerator.php* another tool was developed to visualize sparse matrices in CRS format. Once the database information is entered for the tool, and permissions are setup for the user, the tool becomes usable.

In order to allow the end user to run an MPI program it is compiled and put in the interface's home folder. Information is inserted into the database to 'install' the application in the system. Permissions are applied to an end user to be able to run the program.



Run Application

Application Name: MPI Test

Application ID: 1

Description: Each processor outputs ID and number of total processes in the application.

Number of Processors: 1

Figure 6 Run application screen provides settings for the user such as how many processors to use

The installation of MOWIC is minimal. A MOWIC user is created and assigned to the web server group. The web folder is copied into a web accessible directory. The SQL file included is run in a newly created MySQL database. The three system settings are adjusted in the MySQL database to reflect the location of the MOWIC user's home directory and data directory to be used. A first administrator user must be manually inserted into the database. The password field should be processed through the md5 function. A config.php file is used to code in the username and password needed to connect to the MySQL server. This can be edited in any text editor. These steps complete installation.

4. EXPERIMENTAL EXAMPLES

4.1 Using the System

The goal of the system is to allow parallel programmers to share the fruits of their labor in a more efficient way. In the experimental examples, it can be seen that the interaction with the cluster is much simpler than using a standard console. Though many programs are written specifically for a particular task, others could be shared permanently with the web. Even the applications which are written for single use may be put on the web as verification for research. Visualization tools could be used and reused among many different projects. Doing this will allow for researchers to speed up their research, minimizing the time needed to train research assistants on the particulars of site being used.

4.2 Sample Parallel Code and Output Viewed

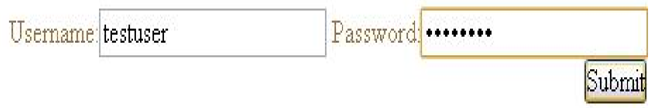
MOWIC is designed to allow a specialized visualization team to be able to develop and deploy visualization software independent of the parallel code, just as in this example. The interface team or person may often provide feedback to the programmers, but has many options available to connect the interface to applications and tools. Such a tool as simply to remove repeating space characters from a text file may be useful and could easily be integrated. As the system gets larger and more robust, it begins to mimic a full desktop operating system, yet yields the computational power of the cluster behind it.

The following explains the steps needed to run code on a cluster and view the resulting file with a visual tool. The particular visual tool in this case is simply a plain text viewer.

1. Open a web browser on a computer connected to the internet

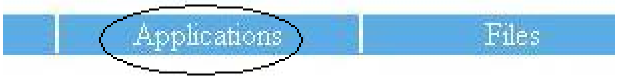
2. Navigate to http://cscluster.kysu.edu/MOWIC

3. Login to MOWIC with username: testuser and password: testpass

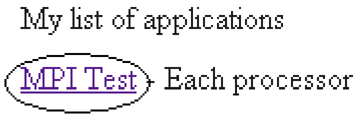


Username: testuser Password: ••••••••

4. Click the ‘Applications’ menu option to list installed applications



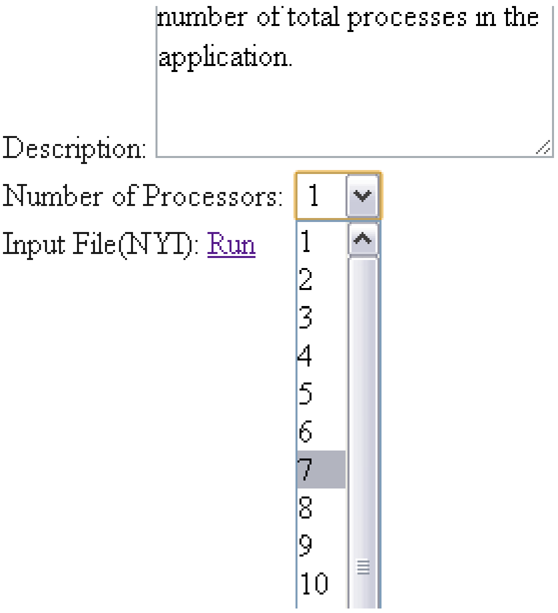
5. Click ‘MPI_Test’ to chose this simple parallel code



My list of applications

MPI Test Each processor

6. Select any choice of number of processors to use



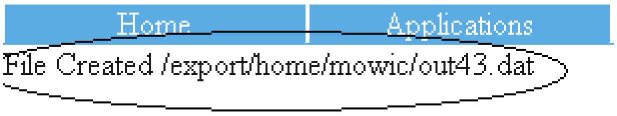
Description: number of total processes in the application.

Number of Processors: 1

Input File(NYI): Run

7. Click ‘Run’ to execute the program on the cluster in parallel on the number of processors selected.

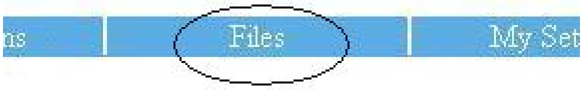
8. Note the filename created



File Created /export/home/mowic/out43.dat

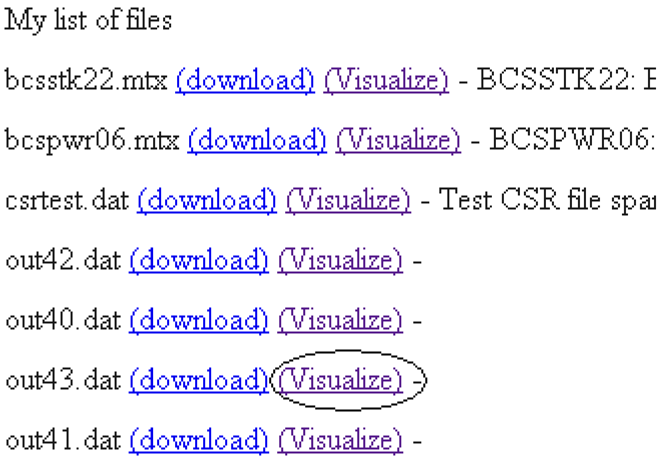
9. Refresh the web page to reload the data from the server

10. Click the ‘Files’ menu option



11. Locate the file noted in step 8

12. Click ‘Visualize’ next to the file that was created



My list of files

bcsstk22.mtx (download) (Visualize) - BCSSTK22: I

bcspwr06.mtx (download) (Visualize) - BCSPWR06:

csrtest.dat (download) (Visualize) - Test CSR file spa

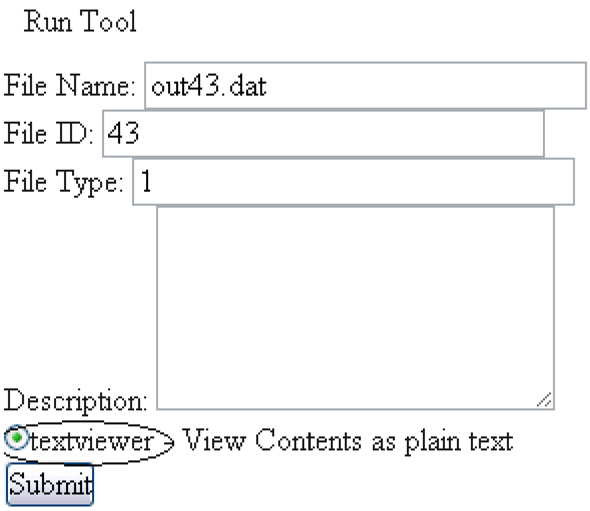
out42.dat (download) (Visualize) -

out40.dat (download) (Visualize) -

out43.dat (download) (Visualize) -

out41.dat (download) (Visualize) -

13. Select the ‘TextViewer’ radio button



Run Tool

File Name: out43.dat

File ID: 43

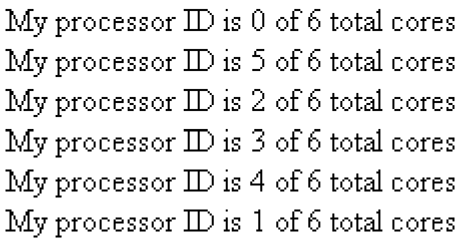
File Type: 1

Description:

● textviewer View Contents as plain text

14. Click submit

15. The file is then displayed as a text file to the user



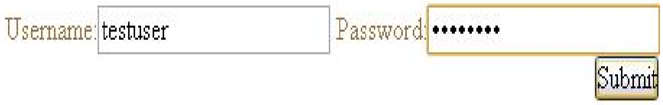
```
My processor ID is 0 of 6 total cores
My processor ID is 5 of 6 total cores
My processor ID is 2 of 6 total cores
My processor ID is 3 of 6 total cores
My processor ID is 4 of 6 total cores
My processor ID is 1 of 6 total cores
```

4.3 Visualize a Data File

1. Open a web browser on a computer connected to the internet

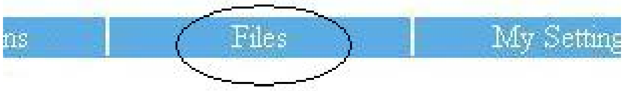
2. Navigate to http://cscluster.kysu.edu/MOWIC

3. Login to MOWIC with username: *testuser* and password: *testpass*

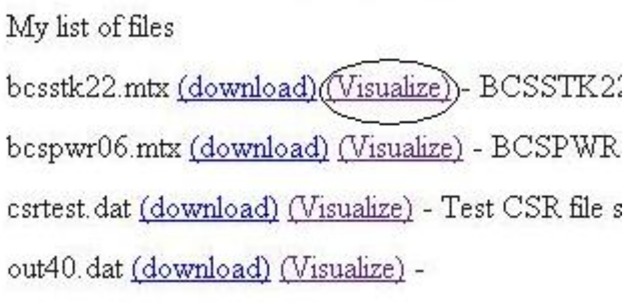


Username: testuser Password: ••••••••

4. Click on the menu option ‘Files’



5. Click ‘Visualize’ on the line for bcsstk22.mt



My list of files

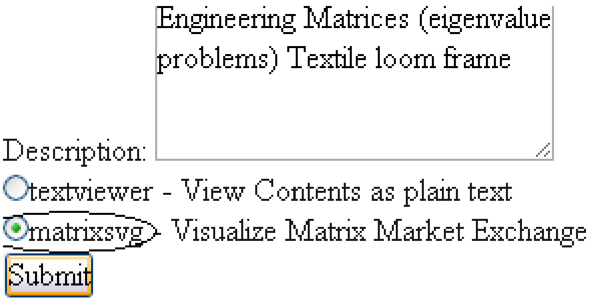
bcsstk22.mtx (download)(Visualize)- BCSSTK2

bcspwr06.mtx (download) (Visualize) - BCSPWR

csrtest.dat (download) (Visualize) - Test CSR file s

out40.dat (download) (Visualize) -

6. Click the radio button next to matrixsvg



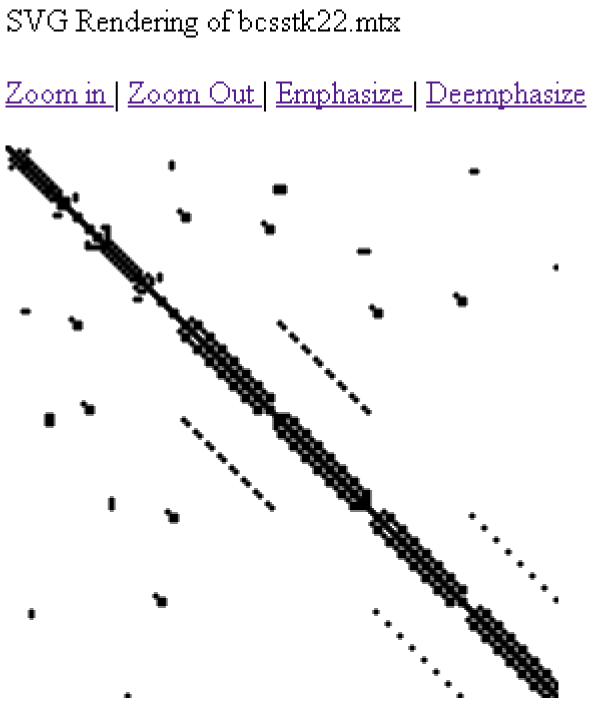
Description: Engineering Matrices (eigenvalue problems) Textile loom frame

○ textviewer - View Contents as plain text

● matrixsvg - Visualize Matrix Market Exchange

7. Click ‘Submit’

8. The matrix is displayed; it may be zoomed in and out, emphasized and deemphasized



SVG Rendering of bcsstk22.mtx

5. HOW TO USE MOWIC

5.1 MOWIC Administration

Administrator of MOWIC should be familiar with the web server and OS of the cluster. MySQL or general SQL knowledge is helpful during the installation process but not absolutely necessary. This requirement can be further mitigated with the use of phpMyAdmin which can easily be installed to provide a visual look into the database if needed. The administrator should be a web programmer capable of working with PHP, XHTML, and JavaScript. Any other experience is useful but not critical. Custom tool development can take advantage of talents with Silverlight, flash, java applets, jQuery, HTML5 canvas, etc. Computational Scientists could continue to develop programs as normal. The tool developer should have no problem quickly connecting applications and tools to the interface.

5.2 MOWIC Application Development

Application development for cluster with a MOWIC interface may function as they regularly do. Output should be directed to *stdout*, standard output stream, which will be redirected to an output file. Parameters should be clearly defined and setup in the installation of the application by the administrator before using. The program currently must write to one output file only. If multiple files are needed to be written to, at the current time, a delimiter can be used and the file split up after completion by a separate program. The current version of MOWIC does not support batch schedulers, and is currently targeting small clusters using *mpirun*. This feature is planned for the next version of the interface.

5.3 MOWIC Visualization Tools

Visualization tools are built as a site within a site. These are installed in the apps directory of MOWIC in their individual folders. The folder name becomes the unique app name of the tool. The tool will be embedded in an iFrame within the interface so only requires navigation related to the tool. As many pages and files as needed may be stored within. This may use standard html design, Ajax [13], with any server side programming required. It may make command line calls to anything in the MOWIC user's directory. This allows for batch files to be written and integrated within them to visualize with Matlab, gnuplot, ParaView, or other tools. Visualizations are left to the designers creativity and may include HTML5 canvas visualizations, flash, Silverlight, svg, etc. The filename to be visualized will be passed in a POST argument named 'filename'. The PHP relational include 'include (../system/global.inc.php)' [14][15] is used to access MOWIC library functions which will be provided upon request.

6. CONCLUSIONS AND FUTURE DEVELOPMENT

HPC use is limited to those with a specific level of computer expertise, access to sufficient support personnel or collaborators. A familiar method to interface to large, complex computer systems has proliferated with the web. In demonstrating the ability for user to login, run specific HPC software with a specific dataset and visualize the output, we have concluded that such a system could greatly reduce the training required to access and utilize HPC in research. An HPC management system, like MOWIC, could be further developed to easily provide a standard interface to HPC resources, independent of their hardware and software implementation. Extensible support for users to develop or incorporate their own visualization tools would allow for virtually any HPC resource to be accessible to researchers with any level of computer proficiency.

Many features are in the pipeline for further development of MOWIC. The nature of such a tool to change cluster interactions so dramatically will require many updates to ensure maximum usefulness and compatibility. Batch schedulers will need to be usable with MOWIC. The system may be enhanced with standard features such as LDAP connections for users to login and access their own account directories. Administrative features such as themes, configuration settings, module installation and un-installation will also be important. Cross-HPC APIs could be developed to allow geographically separate systems to be accessible via a single interface which would provide a cloud-like aspect to the system.

Many visualization tools will need to be developed. An online help system including documentation and video demonstrations should be created online to support the interface. APIs could be created to collect data, allowing remote sensors to feed data directly into the system via the web. Users could have sharing tools. Chat and collaboration tools could be implemented. A mobile version of the site could be created allowing users to view the status of jobs, email the results to collaborators, etc.

7. ACKNOWLEDGMENTS

Thanks are given to Dr. Siddhartha Bhattacharyya and Jerome Walker at Kentucky State University for coordinating IT staff to allow connection of the interface to the Internet as well as John Roberts for his work in exploring visualization techniques. Thanks are given to Dr. Mike Unuakhalu at Kentucky State University for his guidance and support. This research work is support by Kentucky National Science Foundation (NSF) Experimental Program to Stimulate Competitive Research (EPSCoR) - Cyber-Infrastructure Initiative program.

8. REFERENCES

[1] Roskoski, Joann P., et al**.** 2010. Dear Colleague Letter: *Cyberinfrastructure Framework for 21st Century Science and Engineering (CF21).* National Science Foundation http://www.nsf.gov/pubs/2010/nsf10015/nsf10015.jsp.

[2] Beniof, Marc R. and Lazowska, Edward D**.** Computational Science: Ensuring America's Competitiveness. *Networking and Information Technology Research and Development.* [Online] 2005

http://www.nitrd.gov/pitac/reports/20050609_computational/computational.pdf

[3] Software Infrastructure for Sustained Innovation . *National Science Foundation,* [Online] National Science Foundation http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503489&org=NSF&sel_org=XCUT&from=fund

[4] Erwin, D. W. and Snelling, D. F. 2001. *UNICORE: A Grid Computing Environment.* Euro-Par 2001. pp. 825-834.

[5] Wang, Shaowen, Wilkins-Diehr, Nancy and Martin, Stuart. 2009. *Enabling Geosciences Gateways to Cyberinfrastructure,* Computers & Geosciences, pp. 2283-2294, December, 2009

[6] HUBzero Platform for Scientific Collaboration. [Online] http://www.hubzero.org

[7] Schultheiss, B. C. and Baalbergen, E. H. 2001, *Utilizing Supercomputer Power From Your Desktop,* Amsterdam : s.n., 2001. HPCN 2001 Conference.

[8] MySQL Homepage. *MySQL.* [Online] http://www.mysql.com

[9] PHP Manual. [Online] http://php.net/manual/en/index.php.

[10] W3Schools Online Tutorials. [Online] http://www.w3schools.com.

[11] Rocks Clusters Documentation. *Rocks Clusters.* [Online] www.rocksclsuters.org.

[12] Netlib Repository at UTK and ORNL[Online] http://www.netlib.org

[13] Holdener III, Anthony T , 2008. *Ajax The Definitive Guide.* Sebastopol, CA : O'Reilly Media, Inc., 2008.

[14] Williams, Hugh E. and Lane, David, 2004. *Web Database Applications with PHP and MySQL.* Sebastopol, CA : O'Reilly Media, Inc., 2004.

[15] Malan, David J., 2009. *Harvard E-75 Online: Building Dynamic Website.* [Online] 2009. http://cs75.tv/2009/fall/.