PolyRun - Polymer Microstructure Exploration HPC Gateway

Alec Lofquist †
Department of Computer Engineering
Iowa State University
Ames, IA

David M. Ackerman †
Department of Mechanical Engineering
Iowa State University
Ames, IA

Steven Clark
ITaP Research Computing
Purdue University
West Lafayette, IN

Christopher Thompson
ITaP Research Computing
Purdue University
West Lafayette, IN

Amit Chourasia ‡
San Diego Supercomputer Center
University of California San Diego
La Jolla, CA
amit@sdsc.edu

Baskar Ganapathysubramanian ‡
Department of Mechanical Engineering
Iowa State University
Ames, IA
baskarg@iastate.edu

***Abstract*—This paper describes design and development of a gateway enabling polymer scientists with limited HPC background to model the equilibrium microstructures of polymer melts. These microstructures are critical to the physical properties of polymers and are of great interest in tailoring polymer microstructures to meet application challenges. The gateway utilizes a Qt application running in a HubZero framework to allow users to configure, submit, track, and visualize polymer structure simulations. In particular the gateway provides an interface to perform multidimensional parameter sweeps of different configurations to construct phase diagrams which are of high interest to the polymer community. The gateway enables both seasoned and non-hpc users to easily perform complex computations and utilize simulations as an aid in designing experiments towards materials by design.**

I. Introduction and Motivation

Polymers are long chain macromolecules with physical properties that make them appealing for a wide range of uses in structural support, organic electronics, and biomedical applications. The microscopic structure adopted by polymers plays a key role in determining their suitability for advanced applications. Changes in the polymer configuration or processing can control the structure. Physical experiments and theoretical studies are used to determine the structure formed under a given set of conditions. It has long been recognized that theory and simulation can guide experiments to desirable structures. We have previously developed a finite element framework for running these simulations on HPC systems [1]. This framework is powerful and flexible; however, it suffers from accessibility issues common to many traditional HPC software. This learning curve of the command line interface limits the user base of the code. With that in mind, we have developed a user-friendly gateway providing a graphical front-end for the simulation code. The gateway runs a Qt based application within a browser based virtual machine that is portable across computer architectures. This paper briefly describes the goals and development decisions involved in the creation of the gateway.

† co-lead authors. ‡ co-corresponding authors.

Presented at Gateways 2018, University of Texas, Austin, TX, September 25-27, 2018.
https://gateways2018.figshare.com/

II. Design Goals and Decisions

Based on the problem description above, we selected several focus points. First, the target audience consists of two main groups: 1) existing polymer scientists with limited computational background, and 2) new researchers (e.g. undergrads or first year grad students) interested in doing HPC related work. Both are groups who might benefit from polymer science simulations but may be put off by the steep learning curve required to use standard command line based simulation tools.

With the target audience determined, the following main priorities were identified:

- A Graphical User Interface (GUI) that matches what most computer users expect. Like most HPC software, the underlying simulation code is intended to run from a terminal on a remote system. Past experience shows that command line applications are a major hurdle for many users. Without a graphical interface, experimentalists will be hesitant to invest the effort to learn a command line system and new researchers will spend a lot of time learning terminal commands before they can do any actual science.
- Data Management and Visualization: Simulations may generate large amounts of data on the computing resources. Data primarily includes snapshots of structural evolution over time and statistics over the course of the run. This data needs to be summarized in a meaningful and easily accessible way while also being available to the user in full for later use.
- Ready to use with minimal setup: The simulation software depends on external libraries and has multiple setup steps in order to build it. While those can be bundled together in a distribution, it will still be a complicated

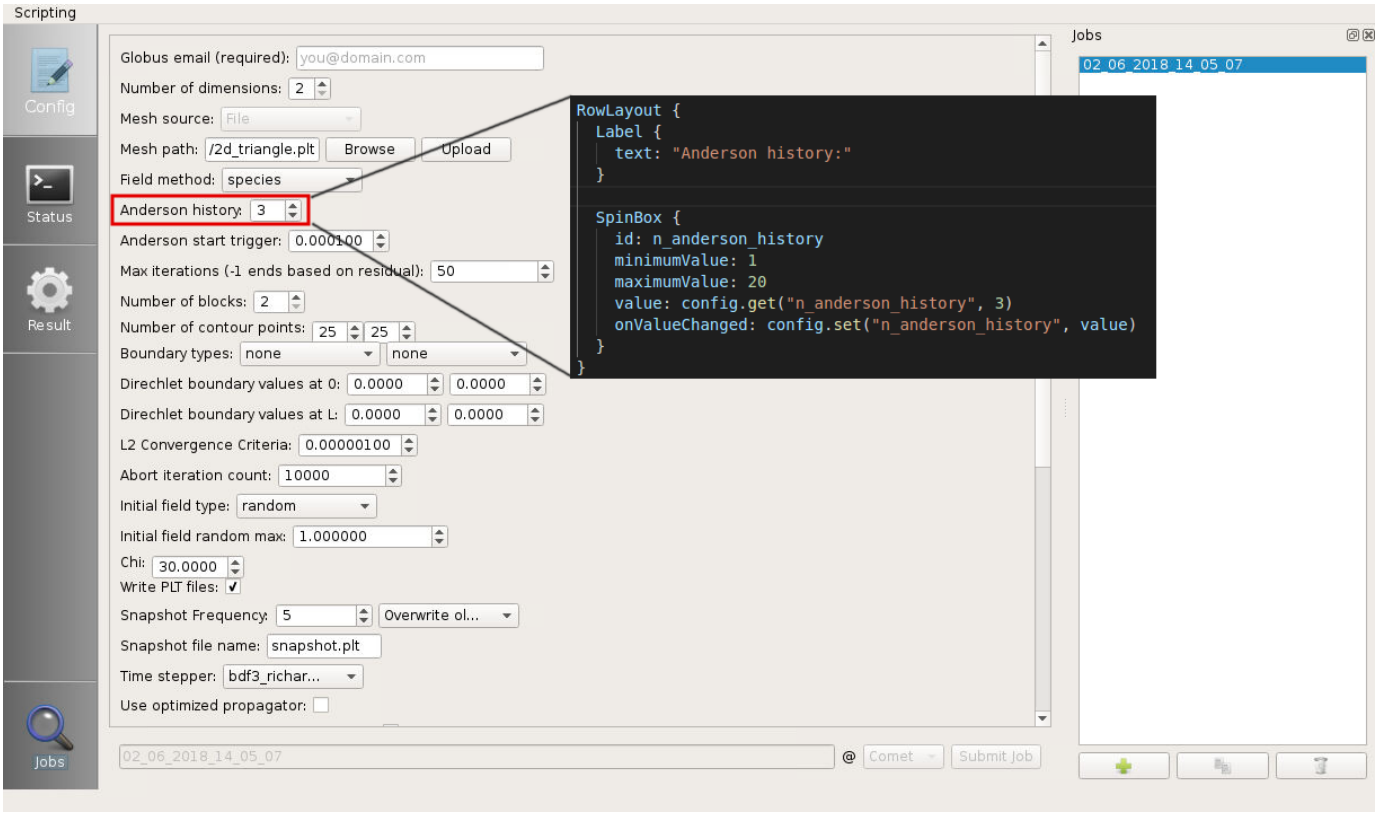

Fig. 1. The job configuration screen with a sample of the QML code used to generate it.

system. In addition, for simulations that will run on HPC resources, there may be significant barriers to overcome in gaining access before starting any calculations. Minimizing set up can get users involved in answering interesting science questions quickly.

- Maintain the power of the HPC software: The polymer science software was designed to support both individual simulation runs as well as broad parameter sweep campaigns. The parameter sweeps are of particular interest to researchers as they allow exploration of the phase space of potential polymer configurations. These sweeps are computationally demanding and require numerous simulation jobs to be submitted at once. If the gateway does not continue to support this capability, its value would be substantially reduced.

All of the development was targeted to meet one of the above goals. The first approach was a Qt-based desktop application designed to manage creation of job scripts. This helped address the first and last priorities and proved successful. Even in a rough form, it was utilized by an undergraduate for his honors research project [2], [3]. Cutting the learning curve by providing a familiar GUI allowed him to get started with the interesting part of generating results without needing to learn Linux terminal commands. Despite that success, he still required help in getting the system set up and support for data sweeps suffered due to issues with job management. Furthermore, despite being written with the cross-platform Qt library, porting the application to Windows and Mac was deemed too time-consuming.

While promising, this approach did not meet the priorities listed above. The research group's focus and expertise is with the science of polymers and the development of the simulation software. Expansion of the GUI to meet the given needs was outside of that area of expertise. To address these needs, assistance was requested through the Extended Collaborative Support Services [4] program of XSEDE [5]. This program provided the required technical expertise in networking, visualization, and virtual machines.

Given limited development time, an appealing approach was to deploy the existing Qt software within DiaGrid [6], a HubZero [7] based virtual machine run by Purdue University. This choice offered several advantages: 1) it preserved existing work that had proven successful, 2) it enabled the use of existing, well tested and well supported HPC job submission and tracking software [8], 3) ongoing development costs were reduced by only needing to support a single platform (the Linux based virtual machine), 4) user setup is limited to creating an account on the DiaGrid system, 5) HPC jobs can be routed through an existing allocation if desired, 6) upgrades automatically propagate for all users, and 7) wide availability via an externally supported host. Several of these features could be met in other ways, but the reduction in upfront development and ongoing support effort made this a compelling approach.

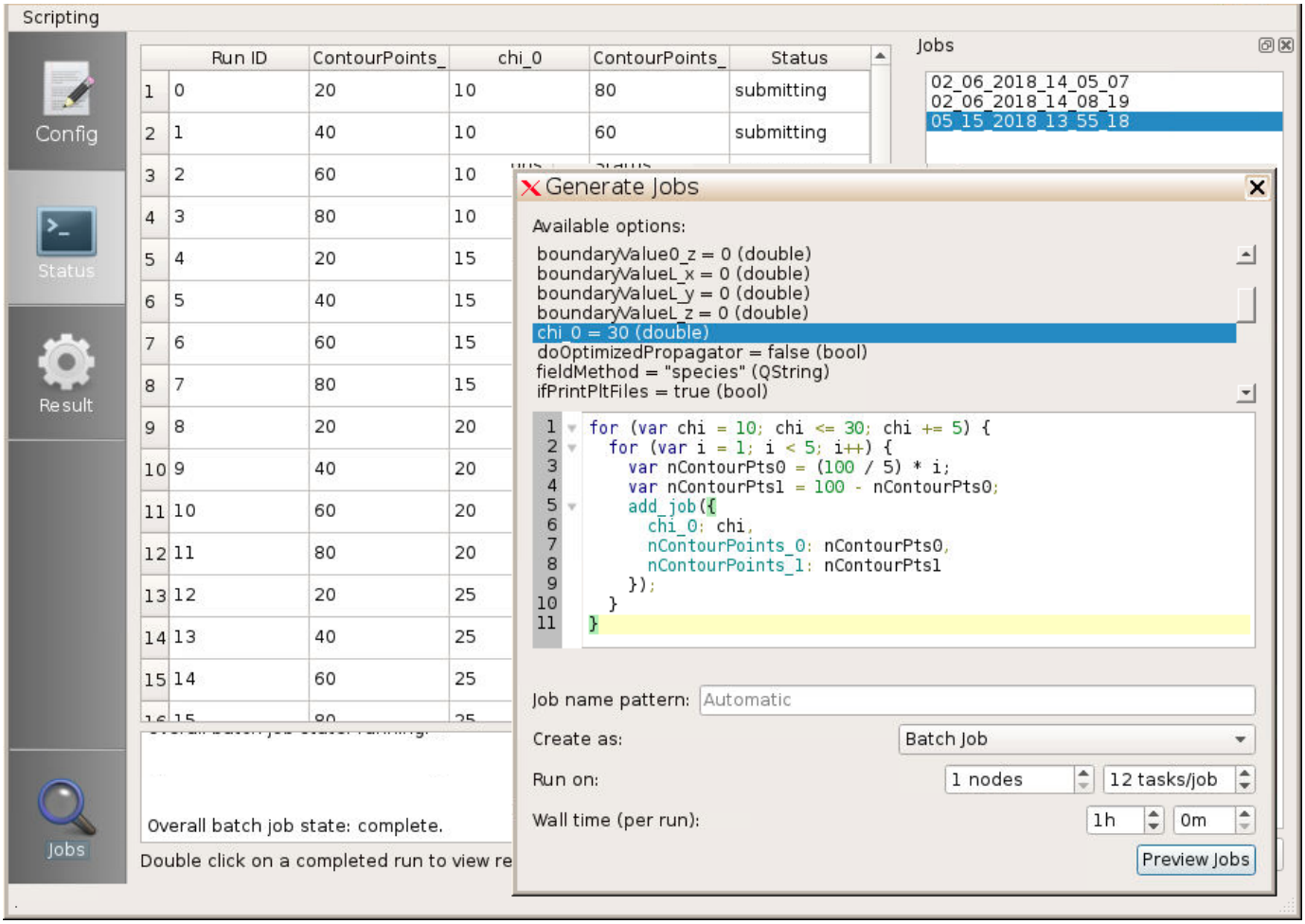

Fig. 2. Job status matrix for parameter sweep of jobs. The inset shows the code used to generate the simulation jobs in the status matrix.

III. Implementation on Virtual Machine

Each of our primary goals, along with the approach to meet it and challenges encountered, is outlined below.

A. Graphical Interface

As mentioned earlier, this is the foundation of the project. The original interface was designed as a stand-alone Qt application. The normal text based configuration was converted to an input form using QML (Fig. 1). Related items are connected via scripting (e.g. changing from 2D to 3D simulations triggers a change in the required number of simulation box dimensions). Jobs are managed through a submission page and a status page provides information on the progress. Once completed, the results can be visualized using snapshots taken during the simulation and plots of data values of interest. Representative images of the intermediate and final system states are generated on the computing resource using VisIt [9] with predefined views that illustrate the results of the simulation. This enables a rapid evaluation of the results and permits the researcher to observe the progress of the simulation. The raw data files may be retrieved from the computing resource through the user interface to enable further processing. Porting the Qt code to the virtual machine required several non-standard Qt libraries to be installed, necessitating updates to the underlying DiaGrid system. A larger issue was limited support for HTML widgets within the supported Qt environment. This prevented the use of HTML and JavaScript (which had been used in rapid prototyping of the user interface components) within the widgets used for layout and display of information.

B. Minimal User Setup

The user is required to create an account on the DiaGrid system, and needs to be given access to an HPC allocation. Beyond that, the virtual machine environment is already set up to run the system. Jobs are run on HPC resources using a community account with a user specified HPC resource billing. This enables users to either utilize an existing allocation which they are granted access to, or to obtain an allocation on their own for their project. The Diagrid system provides the account authorization.

C. Retain HPC Capabilities

A critical component of the simulation software is the ability to submit large batches of related jobs. Without this feature, it

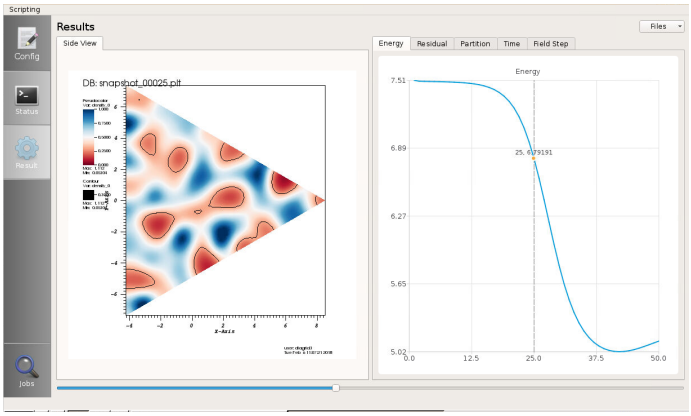

Fig. 3. The results page allowing users to visualize the structures and time evolution of data items of interest.

would not be possible to achieve one of the primary scientific goals: the creation of phase diagrams from large numbers of simulations. A recent project utilizing this code required approximately 5000 simulations jobs, with most grouped into batches of 10-200 jobs that covered a span of parameters. To support this, a batch submission feature is included which can generate jobs with a range of parameters. Internally, the management of large batches of jobs is handled automatically by HubZero's job submission middleware. The user is kept updated on the progress of the batch jobs via a status page as shown in Fig. 2. Job results can be individually inspected via the results page (Fig. 3). The inset shown on the status page is an example of the code used to create the list of batch jobs. While the goal of the project is to minimize the need for users to write any sort of code, it proved unavoidable in this case. A fully graphical means of specifying jobs would require significant development time and be limited to the capabilities provided by the developer – likely requiring additional development effort every time a new use case is needed. The code based generation avoids these issues. To minimize the burden on users arising from this choice, the following strategies are employed: 1) several detailed example cases are available to users in the documentation, 2) the JavaScript language was chosen as it has a large number of freely available tutorials a user can reference, 3) a validation of code is performed with error messages which specify exact line numbers of problems, and 4) users are able to preview the resulting job list prior to accepting it. These strategies are expected to make it feasible for users to develop batch jobs with minimal difficulties.

D. Data Management

This proved to be the most complicated part of development. The fundamental challenge stemmed from the need for users to have access to the large amounts of simulation data generated on a system they, in general, will not have direct access to. Data from a modest size simulation run could reach 100 GB when keeping only final configuration data. Simulations on more complex polymer systems would have larger data production. As shown in Fig. 3, the evolution of the physical structure is of interest. Storing intermediate time points could lead to 100 times more data than keeping only the end results. This is not an unmanageable amount of data but due to the size, it is not feasible to move all the data to the DiaGrid system. It is impractical and unnecessary to automatically move the data to the user's local system. Not all of this is data is ultimately necessary, but the determination of which data is needed is made by the user during analysis. The user will ultimately need access to the data that is deemed relevant. The goal of this system is to help the user filter through the data to decide what is relevant and then make that data available for further analysis. Our approach is two fold. First, we provide representative previews of all data to enable the user to determine which data is needed for future analysis. This is intended to reduce the number of files that need to be transferred. Second, an interface is provided to the Globus data management service [10], [11]. Data created through the simulations is transferred to long term storage on the HPC cluster and automatically made accessible by the GUI tool using the web-based Globus interface. During the simulation setup, the user can provide a Globus email that the results will be shared with. The user is then able to download the specific data files needed. Files can be transfered through Globus to a local system or to another HPC system for further post processing or visualization. File ownership is maintained by the gateway, however access to specfic files is governed by Globus sharing permission which prevents users from accessing data they did not create. Storage of data on the HPC system will be governed by a data retention policy that may adapt over time as the gateway matures.

IV. Conclusion

This tool has been deployed on the DiaGrid system with initial support for the Comet supercomputer at the San Diego Supercomputer Center. The process of porting the Qt-based application to the DiaGrid system presented several challenges due to the nature of running on a virtual machine. The motivation of porting an existing program to the virtual machine was practicality. Utilizing an existing, working system was preferable to creating a new system. This was one of several design trade offs needed to balance user experience and limited development time. Referencing the key goals listed above helped guide the development process to ensure the resulting gateway met its intended function. Future work will focus on growth of a user community and expanding support for the different polymer systems within the software.

Acknowledgments

DMA, AL, and BG were partially supported by NSF Grant No. DMR-1435587 and CMMI-1149365.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562 using the Comet system at the San Diego Supercomputer Center.

REFERENCES

[1] D. M. Ackerman, K. Delaney, G. H. Fredrickson, and B. Ganapathysubramanian, "A finite element approach to self-consistent field theory calculations of multiblock polymers," *Journal of Computational Physics*, vol. 331, pp. 280 – 296, 2017.

[2] J. Green, D. M. Ackerman, and B. Ganapathysubramanian, "Effects of confinement geometry on biblock copolymer systems using finite element analysis," in *Presented during National Conference on Undergraduate Research*, (Memphis, TN), 2017.

[3] J. Green, D. M. Ackerman, and B. Ganapathysubramanian, "Phase behaviour of polymer systems under geometric confinement," *submitted*, 2018.

[4] N. Wilkins-Diehr, S. Sanielevici, J. Alameda, J. Cazes, L. Crosby, M. Pierce, and R. Roskies, "An overview of the xsede extended collaborative support program," in *High Performance Computer Applications - 6th International Conference, ISUM 2015, Revised Selected Papers*, vol. 595 of *Communications in Computer and Information Science*, (Germany), pp. 3–13, Springer Verlag, 1 2016.

[5] J. Towns *et al.*, "Xsede: Accelerating scientific discovery," *Computing in Science & Engineering*, vol. 16, pp. 62–74, Sept.-Oct. 2014.

[6] "Diagrid Web page." https://diagrid.org/.

[7] M. McLennan and R. Kennell, "Hubzero: A platform for dissemination and collaboration in computational science and engineering," *Computing in Science & Engineering*, vol. 12, no. 2, pp. 48–53, 2010.

[8] *Submit Command (HubZero documentation)*. https://help.hubzero.org/documentation/current/tooldevs/grid.submitcmd, 2009.

[9] H. Childs *et al.*, "VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data," in *High Performance Visualization–Enabling Extreme-Scale Scientific Insight*, pp. 357–372, Oct 2012.

[10] I. Foster, "Globus online: Accelerating and democratizing science through cloud-based services," *IEEE Internet Computing*, vol. 15, pp. 70–73, May 2011.

[11] B. Allen *et al.*, "Software as a service for data scientists," *Commun. ACM*, vol. 55, pp. 81–88, Feb. 2012.