Journal of Computational Science 9 (2015) 33–38

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Fast equilibration of coarse-grained polymeric liquids

David Ozog[a,*], Jay McCarty[b], Grant Gossett[b], Allen D. Malony[a], Marina Guenza[b]

[a] University of Oregon, Department of Computer and Information Sciences, Eugene, OR, USA
[b] University of Oregon, Department of Chemistry, Eugene, OR, USA

ARTICLE INFO

Article history:
Available online 20 April 2015

Keywords:
Atomistic simulation
Coarse-graining
Scientific workflows
Polymeric liquids
LAMMPS

ABSTRACT

The study of macromolecular systems may require large computer simulations that are too time consuming and resource intensive to execute in full atomic detail. The integral equation coarse-graining approach by Guenza and co-workers enables the exploration of longer time and spatial scales without sacrificing thermodynamic consistency, by approximating collections of atoms using analytically-derived soft-sphere potentials. Because coarse-grained (CG) characterizations evolve polymer systems far more efficiently than the corresponding united atom (UA) descriptions, we can feasibly equilibrate a CG system to a reasonable geometry, then transform back to the UA description for a more complete equilibration. *Automating* the transformation between the two different representations simultaneously exploits CG efficiency and UA accuracy. By iteratively mapping back and forth between CG and UA, we can quickly guide the simulation towards a configuration that would have taken many more time steps within the UA representation alone. Accomplishing this feat requires a diligent workflow for managing input/output coordinate data between the different steps, deriving the potential at runtime, and inspecting convergence. In this paper, we present a lightweight workflow environment that accomplishes such *fast equilibration* without user intervention. The workflow supports automated mapping between the CG and UA descriptions in an iterative, scalable, and customizable manner. We describe this technique, examine its feasibility, and analyze its correctness.

1. Introduction

Despite considerable advancements in hardware and software technologies for supporting large-scale molecular simulations, computational chemists are confined to simulating relatively small systems compared to most laboratory experiments and real world bulk measurements. This constraint quickly becomes apparent in the simulation of polymer melts, which is important for materials science applications. Polymers are macromolecules that consist of repeating units of monomers (such as CH_2) that form long molecular chains. When simulating polymeric systems, many interesting properties depend on the chain length [20], such as the boiling/melting points, the viscosity, and the glass transition temperature. Even with a small number of molecules, however, it is computationally expensive to simulate chains with more than 10^5 monomers each, which is a reasonable chain length to study. Limited memory space also constrains simulations to 10^6–10^{10} total atoms, even on large allocations of modern supercomputers. Considering that a drop of butane contains approximately 10^{20} atoms suggests that these capabilities do not come close to bulk experiments in the laboratory.

Defining a new representation of the polymer as a chain of soft colloidal particles greatly reduces the amount of information to be collected and controlled, which speeds up the simulation. It is well known that modeling fewer colloidal particles with an appropriate potential decreases the degrees of freedom and computational requirements by an amount proportional to the granularity [3]. The representation of a polymer as a chain of soft blobs also allows the chains to more easily cross each other, decreasing the required time for the simulation to find the equilibrium structure. However, the information on the molecular local scale needs to be restored at the end of the simulation to account for properties on the *united atom* (UA) scale. In a nutshell, it is important to alternate between the *coarse-grained* (CG) representation (which speeds up the simulation) and the UA representation (which conserves the local scale information). By quickly switching back and forth from UA to CG, we open doors to new studies of polymeric systems while maintaining simulation accuracy and efficiency. While optimizing the computational performance of CG codes is important, without a way to incorporate UA-level detail, CG efficiency has relatively less value.

* Corresponding author.
E-mail addresses: ozog@uoregon.edu (D. Ozog), jmccart4@uoregon.edu (J. McCarty), ggossett@uoregon.edu (G. Gossett), mguenza@uoregon.edu (M. Guenza).

http://dx.doi.org/10.1016/j.jocs.2015.04.015
1877-7503/

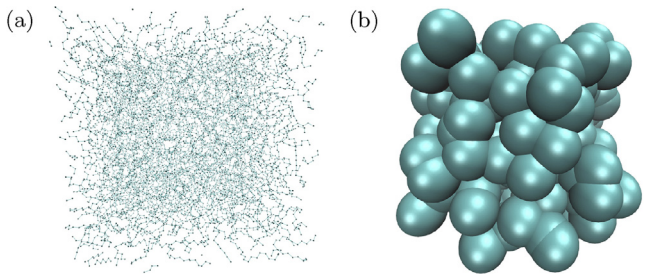

Fig. 1. UA versus CG descriptions. CG spheres appear hard, but are soft with long range effects. (a) UA representation with 80 chains 120 monomers per chain. (b) The corresponding CG representation with 3 sites per chain (and 40 internal monomers per site).

Therefore, this paper focuses on integrating an approach for conducting simulations that exploit both CG efficiency and UA accuracy (Fig. 1).

Unfortunately, it is not trivial to automate the transformation between the CG and UA representations for general sets of input parameters. Accomplishing the overall task involves multiple processing steps, with different applications, programming languages, naming schemes, and data representations. This issue is common to many other scientific software environments and is collectively referred to as the scientific workflow problem [10]. In short, the collection of all the required steps to conduct an overall research study comprises a *workflow* that may consist of several large simulations, real-world experiments, human intervention and analysis, and more. In this paper, we present a lightweight and customizable approach for creating an effective scientific workflow in the context of our CG/UA simulation experiments. We discuss techniques that are general to other computational problems and explore new ideas that are not prevalent in other more heavyweight workflow toolkits.

The paper is organized as follows: Section 2 contains background information regarding the CG approach and integral equation theory, Section 3 discusses the design and implementation of the fast equilibration workflow, Section 4 presents our experiments in evaluating the workflow, Section 5 discusses related work, and Section 6 contains concluding remarks.

2. Background

This section briefly reviews the coarse-grained approach based on integral equation theory. We consider a homopolymer fluid (in which all monomers are the same type) with molecular number density, ρ_m, consisting of n polymer chains. Some number of monomer sites, N, makes up each polymer chain, where each site is usually taken to be either a CH, CH_2 or CH_3 group. This bead-spring description is the UA simulation model. Within the UA description, the Polymer Reference Inter Site Model (PRISM) site-averaged Ornstein-Zernike equation relates the relevant pair correlation functions in Fourier Space [21]:

$$\hat{h}^{mm}(k) = \hat{\omega}^{mm}(k)\hat{c}^{mm}(k)[\hat{\omega}^{mm}(k) + \rho\hat{h}^{mm}(k)] \quad (1)$$

where the "mm" superscript denotes monomer–monomer interactions and $h^{mm}(k)$ is the Fourier transform of the total correlation function, $h^{mm}(r)$. In fact, $h^{mm}(r) = g^{mm}(r) - 1$, where $g(r)$ is the well known radial distribution function. Also from Eq. (1), $c^{mm}(k)$ is the direct correlation function, $\omega^{mm}(k)$ is the intra-chain structure factor, and ρ is the monomer site density, given as $\rho = N\rho_m$. The CG representation can be fully represented as a function of the physical and molecular parameters that define the atomistic description. The key quantity to be solved in the CG representation is the potential, which must be included as an input to the molecular dynamics (MD) simulation of the polymer melt in the reduced CG representation. This potential has been derived analytically using an integral equation formalism. In the CG model, each chain contains an arbitrary number n_b of chemically identical blocks, or "blobs". Each block contains a sufficiently large number of sites so that we can utilize Markovian properties of the internal chain's random walk. The integral equation coarse-graining approach (IECG) allows us to determine an analytical solution to the potential when $N/n_b = N_b \approx 30$. At that scale the structure of the chain follows a random walk, and the distribution of the CG units along the chain is Markovian, so that the IECG formalism can be solved analytically [6]. In a macromolecular liquid, the random distribution of the CG units is a general property of any molecule when sampled at large enough lengthscales [8].

The IECG model conserves thermodynamics while accurately reproducing the structure of polymer liquids across variable levels of resolution [6,7,16]. Insights from the IECG theory provide reasonable justifications for some of the advantages and shortcomings of coarse-graining methods in general [13,14].

The potential is solved analytically in the mean spherical approximation, which is valid for low compressible polymer liquids [12]. In the limit of large separations in real space, where $r \gg 1$ (in units of the polymer size), the potential is approximated as [5,7]

$$V^{bb}(r) \approx k_B T \left[\left(\frac{45\sqrt{2}N_b\Gamma_b^{1/4}}{8\pi\sqrt{3}\sqrt[4]{5}\rho_m R_{gb}^3} \right) \frac{\sin(Q'r)}{Q'r} e^{-Q'r} - \left(\frac{\sqrt{5}N_b}{672\pi\rho_m\Gamma_b^{1/4}R_{gb}^3} \right) \left[(13Q^3(Q'r-4))\cos(Q'r) + \left(\frac{945+13Q^4}{\Gamma_b^{1/4}} \right) r\sin(Q'r) + \frac{945r}{\Gamma_b^{1/4}}\cos(Q'r) \right] \frac{e^{-Q'r}}{Q'r} \right], \quad (2)$$

where $Q' = 5^{1/4}\sqrt{3/2}\Gamma_b^{-1/4}$ and $Q \equiv Q'\Gamma_b^{1/4}$. The key quantity of interest here is the universal parameter $\Gamma_b = N_b\rho|c_0|$, which is defined once we decide the level of coarse-graining, N_b, as well as the molecular and thermodynamic parameters of our system. This quantity also depends on the direct correlation function at $k = 0$, c_0, which is in principle not known. However, this function relates to the potential between atomistic units and the isothermal compressibility of the liquid, so it can be determined numerically or from experiments.

When compared with full atomistic simulation (i.e., UA) the CG simulations that use the potential of Eq. (2) show quantitative consistency in the structure and thermodynamics. The CG potential is in this way fully transferable, and it can be conveniently applied in MD simulations of polymer melts, at the chosen thermodynamic and molecular parameters, with computational gain. The ability of CG models to maintain thermodynamic and structural properties while varying the coarse-graining resolution is important when one develops computational techniques with variable resolution. It allows the computational time of the MD simulations to be controlled by changing the resolution as needed. This notion motivates the development of a workflow that supports transformation between the UA and CG descriptions, which dynamically adjusts CG resolution at desired simulation times and spatial regions, while maintaining simulation accuracy and equilibrium through transitions to united-atom simulation.

3. Fast equilibration workflow

This section describes the fast equilibration workflow, which consists of a series of computational programs and analyses that comprise an overall application for quickly stabilizing a polymeric

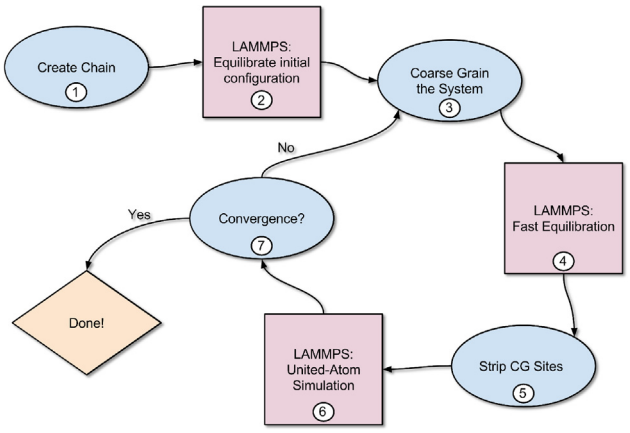

Fig. 2. The fast equilibration workflow. Circles (1, 3, 5, and 7) represent stages of custom programs that either generate coordinates and potentials, transform data for input into LAMMPS, or conduct convergence analysis. Squares (2, 4, and 6) represent parallel LAMMPS molecular dynamics simulations executed via MPI. Our workflow system automates this process for a given set of input parameters. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

liquid. Fig. 2 shows the 7 high-level stages involved in the workflow, each of which may involve multiple processing steps. Each step is accomplished by one or more programs, applications, or simulations. Before this work, these steps each required manual intervention by a researcher, but now they are automated by our fast equilibration workflow.

3.1. Workflow design

Stages 1 and 2 from Fig. 2 initialize the workflow by generating a geometric configuration and equilibrating the system just enough to remove instabilities. Stage 1 randomly generates a polymer system of n chains, each with N monomers. Each chain begins at a random coordinate, and the subsequent N monomers bond randomly onto the enclosing spherical surface. Given a desired n, N and ρ, we generate a collection of random chains within a simulation box of volume V with periodic boundary conditions and length, L, such that $V = L^3$. We also include periodic boundary image flags to help reconstruct the polymer chains later. Because randomly generated configurations likely contain regions of high strain and numerical instabilities due to overlapping chains, we carefully adjust the configuration of these areas. Stage 2 accomplishes this with a brief LAMMPS[1] simulation in which chains slowly drift apart via a soft repulsive potential for 10 ps of simulation time. Then, we minimize energy via a Lennard-Jones potential with a series of short executions, in which the system runs for 1000 timesteps with an incrementally increasing amount of time per step (e.g., 0.02, 0.08, 0.25 fs, etc).

Stages 3 through 7 constitute an iterative strategy that alternates between the UA and CG representations towards equilibration. Stage 3 determines the center of mass for each soft-sphere that encompasses a group of N_b monomers. The center of mass coordinates are the "fictitious sites" of the soft colloids from which the multi-block potential is derived. During this stage we also store the internal monomer configurations within each block for rigid tracking within the upcoming CG simulation (see Section 3.2.2). The multi-block potential, which depends on parameters such as temperature, density, and number of blocks per chain, is generated at runtime during the first iteration [6]. The ensuing LAMMPS execution in Stage 4 simulates the CG system with the multi-block potential, treating the internal monomer chains within a block as coupled rigid bodies. In our experiments below, Stage 4 runs approximately 60,000 timesteps at 3 fs per step, although this is easily customizable.

After the CG simulation, Stage 5 restores the UA description by applying the saved internal chain coordinates to their new location within each updated block position. To mitigate any unphysical bond lengths within chains, Stage 6 runs a short simulation within the UA description using the same Lennard-Jones pair potential from Stage 2. Finally, in Stage 7, we determine whether or not to perform another iteration. While there are several alternatives for evaluating convergence effectively, we choose to consider the total correlation function, $h^{mm}(r)$, because of its relevance in deriving system thermodynamic properties (see Section 4.2). Finally, if we determine that the system has satisfactorily equilibrated, the workflow is complete.

3.2. Workflow implementation

This section describes in more detail the implementation[2] of the workflow design discussed in the previous section. Our overall approach involves bundling each set of Fortran and C programs that encapsulate Stages 1, 3, 5, and 7 (in Fig. 2) with Python scripts. Then, using a standard Python argument parsing system, `argparse`, to define all the parameters in Table 1 at launch time, we subsequently pass the parameter bundle (as a Python object) through the entire workflow program. During initial development of the workflow, the computational and analytical programs were hard coded to contain the parameters, filenames, and directory paths. In addition, no record of launch configurations were stored (such as number of MPI processes, the cluster name, or execution time). After easily restructuring the internal programs and procedures to extract relevant values of the parameters from the `argparse` interface, a fully automated system for doing UA/CG polymer equilibration is in place. For instance, one can now launch the entire workflow with a single command:

```
fast_equil -nchain 350 -nmonomer 192 -
sitesperchain 6 -temperature 450
-bondlength 1.55 -mono-mass 14.0 -cnot -9.7 -
timestep 1.0 ...
```

Certain flags, such as `-viz`, can provide outlets for user analysis by stopping the workflow and presenting a visualization of a desired quantity, such as $h^{mm}(r)$, or the simulation box itself. If flags are not specified, they are set to default values as disclosed by the `-help` reference flag.

Upon each launch of a workflow instance, the parameter set, execution timestamp, and path to output datafiles are saved to a database management system (DBMS) for historical reference and provenance. Our current implementation interfaces the Python workflow runtime code with a local SQL-based DBMS. Future work will consider how to extend this feature towards collaborative efforts by exploring remote database interactions and knowledge awareness/extraction.

The previous discussion is related to defining the parameter space within the various components of the workflow in an automated fashion. However, there are some peculiarities in the LAMMPS side of the workflow in Stages 2, 4, and 6 that deserve special attention. This is the subject of the following subsections.

[1] LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) is a classical molecular dynamics code. http://lammps.sandia.gov.

[2] The source code is available at https://github.com/davidozog/Fast-Equilibration-Workflow.

Table 1
Input parameter dependencies for the stages of UA/CG equilibration. Our workflow system obviates tracking dependencies by implicitly passing all parameters throughout each stage, forming a unique parameter space for each overall experiment. (Items marked with * are not directly passed to LAMMPS, yet they determine the total number of input atoms.)

Input parameter	Create chain	Coarse grain		Strip CG	Converge	LAMMPS
	`create_chain`	`cg_chain`	`multiblock`	`strip_cg`	`hmmr`	`lmp_run`
ρ (density)	Yes	No	Yes	No	Yes	No
T (temperature)	No	No	Yes	No	No	Yes
n (# of chains)	Yes	Yes	No	Yes	Yes	Yes*
N (monomers)	Yes	Yes	Yes	Yes	Yes	Yes*
n_b (# of blocks)	No	Yes	Yes	Yes	No	Yes*
c_0 (dc constant)	No	No	Yes	No	No	No
L (box length)	No	Yes	Yes	No	Yes	Yes
p (MPI procs.)	No	No	No	No	No	Yes
Experiment ID	No	No	No	No	No	No
...						

3.2.1. Generating LAMMPS input and the multi-block potential at runtime

Our initial implementation of the multi-block potential utilizes the `table` feature of LAMMPS for doing bond, angle, and intermolecular calculations. Although a direct evaluation of the potential may feasibly achieve better performance, the `table` approach establishes the overall workflow and, if necessary, is substitutable in future versions. Because the multi-block potential depends on the parameters ρ, T, N, n_b, c_0, and L, we generate the potential at runtime. Fortunately, this takes less than 5 s on a standard processor, which is negligible compared to a typical cycle of the workflow, which can take several hours even on hundreds of processors.

Another interesting feature of our workflow environment is that we generate LAMMPS input files via a template processor system. Originally, template processors were designed to be used to create dynamic websites in which content is generated on-the-fly by filling an HTML template file with runtime data [15]. This avoids the error-prone, tedious, and inflexible approach of generating content with code in this manner:

```
print("<html>\n <body>\n <p>" + str(mycontent) + "</p>\n </body>\n </html>\n")
```

Template processing, on the other hand, renders the following `index.html` input template:

```
<html><body><p>@user.name: @user.age</p></body></html>
```

with standard programming objects like so:

```
template.render("index.html", {"user": Jane})
```

An intriguing analogy existed when we initially generated LAMMPS input files,

```
print("minimize"+ str(etol) +" "+ str(ftol) +" "+ str(maxiter) +...)
```

So, with a LAMMPS input template (`in.lammps`),

```
minimize @etol @ftol @maxiter @maxeval...
```

we can render the simulation input files in the same fashion:

```
parameters = {"etol": 1e-4, "ftol": 1e-6, "maxiter":100, "maxeval": 1000}
template.render("in.lammps", parameters)
```

Template processing is central to the popular *model-view-controller* (MVC) architectural pattern. It is clear that many scientific workflows fit the same design pattern, where the data is the model, the workflow is the controller, and the simulation input/output is the view. Just as the MVC paradigm emphasizes a separation of concerns between database engineers, backend programmers, and designers [19], we see MVC applying equally as well to the data scientists, software engineers, and domain specialists that represent the stakeholders in any effective scientific workflow. It is prudent to note the ubiquity of managing scientific simulations through input files (LAMMPS, GROMACS, NWChem, Gaussian, GAMESS, OpenMC, Nek5000, and many more), and that controlling the parameters dynamically with a clear separation of implementation concerns exposes new research possibilities and opportunities for productivity.

3.2.2. Transforming from the CG to the UA description with the POEMS library

Our approach for transforming between the UA and CG descriptions involves storing the coordinates internal to blocks before the CG simulation and treating them as rigid bodies. This is done using the *Parallelizable Open Source Efficient Multibody Software* (POEMS) library included with LAMMPS. In POEMS, when computing the total force and torque on rigid bodies, the coordinates and velocities of the atoms update such that the collection of bodies move as a coupled set [1]. When transiting from UA to CG, the definition of the rigid blocks and their internal atomic identities occurs in Stage 3 of the workflow simultaneously with the generation of the CG description. When mapping from CG back to UA in Stage 5, the final state of the simulation from Stage 4 is used to restore the rigid coordinates. After stripping the fictitious sites from the simulation box, Stage 6 commences with the appropriate timestamp update to keep the overall simulation time consistent.

In our simulations, we observe that calculating the force and torque on rigid bodies comes at a performance cost. However, more efficient methods for carrying the rigid coordinates exist, for instance, by either (1) ignoring rigid body calculations or (2) regenerating random internal configurations between stages of the workflow. Because our approach only requires that chains' random walks be Markovian, approach 2 may bear fruit. Our preliminary studies show that simulations applying methods 1 and 2 exhibit speedups over UA on the order of the granularity factor. Future work will consider in detail the performance of alternative methods for managing the internal configurations.

4. Experimental

4.1. Experimental setup and evaluation

We conducted our scientific workflow evaluation on the ACISS cluster located at the University of Oregon. Experiments are run on the 128 generic compute nodes, each with 12 processor cores per node (2x Intel X5650 2.67 GHz 6-core CPUs) and 72 GB of memory per node. This is a NUMA architecture with one memory controller per processor. ACISS employs a 10 gigabit Ethernet interconnect that connects all compute nodes and storage fabric. The operating system is RedHat Enterprise Linux 6.2, and MPICH 3.1 is used with the `-O3` optimization flag.

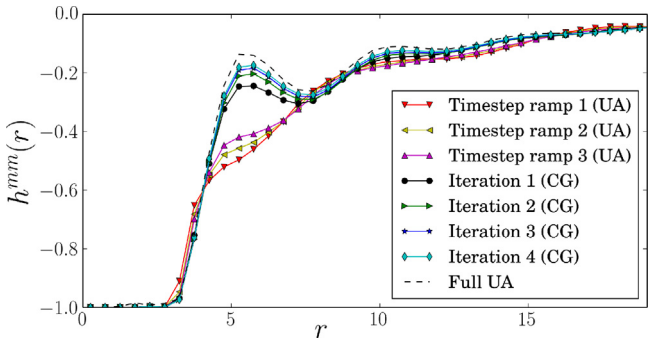

Fig. 3. The total correlation function, $h^{mm}(r)$, for a workflow experiment with $\rho = 0.03355$ sites/Å^3, $T = 450$ K, $n = 350$, $N = 192$, $n_b = 32$, $c_0 = -9.67344$, $L = 126.05$ Å, and $p = 96$ MPI processes on the ACISS cluster. The bottom curves show 3 ramping steps from Stage 2 of the workflow (as in Fig. 2) and the top curves show 4 iterations through Stages 3–7. Satisfactory convergence occurs (both in terms of $\Delta h_i^{mm}(r)$ and w.r.t. full UA) after 4 workflow iterations.

Overall, the overhead introduced by stages 1, 3, and 5 is negligible (far less than 1% of the total workflow execution time). Due to space limitations, we defer computational performance measurements for another publication. We instead focus on the correctness and validity of our approach as verified by the radial distribution function.

4.2. Radial distribution function analysis

Analysis of the radial distribution function is critically important in the evaluation of the validity of a simulation result. Given this function and assuming pairwise additivity, all the thermodynamic properties of the liquid may be calculated [17]. This function, often called $g(r)$, is defined as

$$g(r) = \frac{1}{\rho}\langle \frac{1}{N}\sum_{i}^{n}\sum_{j\neq i}^{n}\delta(\vec{r} - \vec{r}_{ij})\rangle$$

and $h(r) = g(r) - 1$ (we introduced $h(r)$ in Section 2). We calculate $h(r)$ at the convergence step of each workflow iteration, and compare it to the previous determination of $h(r)$. If the mean percentage error is less than a user-defined threshold, then the workflow completes. If available, we can alternatively compare to a long-running "full UA" simulation (with no CG). To clearly specify that we calculate this quantity in the UA (monomer) representation (*not* including the block sites), we henceforth use the *mm* superscript, denoting this function as $h^{mm}(r)$.

Fig. 3 shows $h^{mm}(r)$ for several phases of the workflow. In this experiment, the full UA "gold standard" ran for 1.25 ns of simulation time, and is shown as the black dotted line. The 4 CG iterations were run for a total of 0.72 ns, and each UA transition stage (Stage 6 from Fig. 2) ran for 0.0125 ns. The "Timestamp Ramp" steps correspond to Stage 2 with total simulation times of 20, 80, and 250 fs, respectively. The figure clearly shows that the system converges quickly towards the correct liquid character.

Without the automated workflow, creating Fig. 3 would have required too many human hours and too much tedious intervention between simulations to have been practical. Furthermore, experiments of this nature are easy to launch with different input parameters through a job scheduler, such as PBS. It is no more difficult to conduct an experiment with a few simulations than with hundreds of simulations, except for the time it takes to execute.

5. Related work

Related research supports the notion that the CG representation equilibrates more quickly than fully atomistic replicas, and that re-expressing a melt in the UA description can more completely equilibrate the system [3]. While our verification of $h^{mm}(r)$ is encouraging for verifying correctness, this function is known to be relatively insensitive to certain geometric flaws, particularly in bonding. For instance, it may be possible to have a well-matched $h^{mm}(r)$, but still have occasional bond lengths that are unphysical. Instead, Auhl [2] evaluates $\langle R^2(n)\rangle/n$ and shows that this phenomenon may occur and can be fixed. Future work will examine $\langle R^2(n)\rangle/n$ in Stage 7 of the workflow (which may be less expensive to compute than $h^{mm}(r)$).

Research and development on scientific workflows is pervasive. Some of the more popular frameworks include Kepler (with recent notable applications in drug design [11] and support of distributed data-parallel patterns, such as MapReduce [22]) Pegasus [9], and Taverna [18]. Other related work focuses on the intricacies of data modeling and dataflow in scientific workflows [4]. Instead of committing to a full-blown workflow framework up-front, this paper has focused on the advantages of using a considerably lightweight system for managing input parameters while benefiting from simple data provenance and template processing. As far as we are aware, template processing capabilities within other scientific workflows is limited, and is not to be confused with reusable "workflow templates", which is a powerful plug-and-play concept found in any good workflow suite. Our future work may consider porting our implementation to a more powerful workflow system, but we are so far content with the portability, ease of customization, lack of a GUI, and template processing features within this work.

6. Conclusion

Coarse-graining methods benefit from reduced computational requirements, which expands our capabilities towards simulating systems with a realistic number of atoms relative to laboratory bulk experiments. However, without integrating local scale UA information, configurations cannot equilibrate as completely. This paper focuses on a workflow based solution for exploiting CG efficiency and UA accuracy by iterating between UA and CG representations towards convergence. By utilizing a lightweight scheme for propagating parameters, controlling simulation input files via template processing, and generating multi-block potentials in the CG description, we have constructed an automated system for conducting large scale experiments of polymer liquid systems. We have verified the correctness of the approach of mapping between the atomic and coarse-grained descriptions by comparing the radial distribution function of a long-running atomic simulation with a series of iterations through the workflow.

The fast equilibration workflow system enables the running of experiments that were previously not possible, such as parameter sweeps across different densities, CG granularities, temperatures, and more. Our simple workflow system is based on Python, and incorporates support for parallel (MPI) simulations, data provenance, and template processing. Future work will examine and optimize the computational efficiency, introduce more powerful workflow features, and explore more thorough verifications of correctness.

Acknowledgments

D. Ozog is supported by the U.S. Department of Energy (DOE) Computational Science Graduate Fellowship program under contract DE-FG02-97ER25308. Research at the Univ. of Oregon is supported by DOE, Office of Science, under contracts DE-FG02-07ER25826, DE-SC0001777, and DE-FG02-09ER25873; and the National Science Foundation under CHE-1362500.

References

[1] K. Anderson, R. Mukherjee, J. Critchley, J. Ziegler, S. Lipton, POEMS: Parallelizable Open-source Efficient Multibody Software, Eng. Comput. 23 (1) (2007) 11–23.

[2] R. Auhl, R. Everaers, G. Grest, K. Kremer, S. Plimpton, Equilibration of long chain polymer melts in computer simulations, J. Chem. Phys. 119 (24) (2003) 12718–12728.

[3] J. Baschnagel, K. Binder, P. Doruker, A. Gusev, et al., Bridging the gap between atomistic and coarse-grained models of polymers: status and perspectives, in: Viscoelasticity, Atomistic Models, Statistical Chemistry, volume 152 of Advances in Polymer Science, 2000, pp. 41–156.

[4] A. Bender, A. Poschlad, S. Bozic, I. Kondov, A service-oriented framework for integration of domain-specific data models in scientific workflows, 2013 International Conference on Computational Science (ICCS 2013) 18 (2013) 1087–1096.

[5] A. Clark, J. McCarty, I. Lyubimov, M. Guenza, Thermodynamic consistency in variable-level coarse graining of polymeric liquids, Phys. Rev. Lett. 109 (October) (2012) 168301.

[6] A.J. Clark, M.G. Guenza, Mapping of polymer melts onto liquids of soft-colloidal chains, J. Chem. Phys. 132 (044902) (2010).

[7] A.J. Clark, J. McCarty, M.G. Guenza, Effective potentials for representing polymers in melts as chains of interacting soft particles, J. Chem. Phys. 139 (124906) (2013).

[8] E. David, K. Schweizer, Integral equation theory of block copolymer liquids. II. Numerical results for finite hard-core diameter chains, J. Chem. Phys. 100 (1994) 7784.

[9] E. Deelman, G. Singh, M.-H. Su, et al., Pegasus: a framework for mapping complex scientific workflows onto distributed systems, Sci. Program. 13 (July (3)) (2005) 219–237.

[10] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, et al., Examining the challenges of scientific workflows, IEEE Comput. 40 (12) (2007) 26–34.

[11] U.I. Pek, S. Jesper, L.V. Prasantha, et al., Progress towards automated Kepler scientific workflows for computer-aided drug discovery and molecular simulations, 2014 International Conference on Computational Science (ICCS 2014) 29 (2014) 1745–1755.

[12] C.N. Likos, A. Lang, M. Watzlawek, H. Löwen, Criterion for determining clustering versus reentrant melting behavior for bounded interaction potentials, Phys. Rev. E 63 (February) (2001).

[13] I.Y. Lyubimov, M.G. Guenza, Theoretical reconstruction of realistic dynamics of highly coarse-grained cis-1,4-polybutadiene melts, J. Chem. Phys. 138 (March (12)) (2013) 120000.

[14] I.Y. Lyubimov, J. McCarty, A. Clark, M.G. Guenza, Analytical rescaling of polymer dynamics from mesoscale simulations, J. Chem. Phys. 132 (June (22)) (2010) 224903.

[15] D.-A. Manolescu, M. Voelter, J. Noble, Pattern Languages of Program Design 5, volume 5, Addison-Wesley Professional, 2006.

[16] J. McCarty, A.J. Clark, J. Copperman, M.G. Guenza, An analytical coarse-graining method which preserves the free energy, structural correlations, and thermodynamic state of polymer melts from the atomistic to the mesoscale, J. Chem. Phys. 140 (204913) (2014).

[17] D.A. McQuarrie, Statistical Mechanics, University Science Books, 2000.

[18] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, et al., Taverna: a tool for the composition and enactment of bioinformatics workflows, Bioinformatics 20 (2004) 3045–3054.

[19] T.J. Parr, Enforcing strict model-view separation in template engines, in: Proceedings of the 13th International Conference on World Wide Web, WWW '04, ACM, 2004, pp. 224–233.

[20] M. Rubinstein, R.H. Colby, Polymers Physics, Oxford, 2003.

[21] K.S. Schweizer, J.G. Curro, PRISM theory of the structure, thermodynamics, and phase transitions of polymer liquids and alloys, Adv. Polym. Phys. 116 (1994) 319–377.

[22] J. Wang, D. Crawl, I. Altintas, A framework for distributed data-parallel execution in the Kepler scientific workflow system, 2012 International Conference on Computational Science (ICCS 2012) 9 (2012) 1620–1629.