Polymer Dynamic Field Theory on Graphics Processing Units

To cite this article: I Wright and R A Wickham 2010 J. Phys.: Conf. Ser. 256 012008

View the article online for updates and enhancements.

Related content
- Mapping the MPM maximum flow algorithm on GPUs
  Steven Solomon and Parimala Thulasiraman
- Letter to the Editor
  Makiko Nonomura, Kohtaro Yamada and Takao Ohta
- Fast polynomial multiplication on a GPU
  Marc Moreno Maza and Wei Pan

Recent citations
- Nucleation of the lamellar phase from the disordered phase of the renormalized Landau-Brazovskii model
  Michael F. Carilli et al
- Truncation-based energy weighting string method for efficiently resolving small energy barriers
  Michael F. Carilli et al
- Comparison of Pseudospectral Algorithms for Field-Theoretic Simulations of Polymers
  Debra J. Audus et al
Polymer Dynamic Field Theory on Graphics Processing Units

I. Wright
Perimeter Institute, 31 Caroline Street North, Waterloo, Ontario, Canada N2L 2Y5
E-mail: iwright@perimeterinstitute.ca

R. A. Wickham
Department of Physics, University of Guelph, Guelph, Ontario, Canada N1G 2W1
E-mail: rwickham@uoguelph.ca

Abstract.
This paper explores the application of graphics processing units (GPUs) to a study of the dynamics of diblock copolymer (DBCP) melts. DBCP melts exhibit ordered mesophases with potential applications in nanotechnology, but are often found out of equilibrium. The length scales involved have previously rendered the numerical modelling of these materials intractable in certain regimes. We adapt a code originally written in parallel using the Message Passing Interface to GPUs using the NVIDIA® CUDA™ architecture. We gain a factor of 30 performance improvement over the original code at large system size. We then use this performance improvement to study DBCP melts in two computationally time-intensive regimes: droplet nucleation close to phase coexistence, and dynamics under confinement in a small cylindrical nanopore.

1. Diblock Copolymers
A diblock copolymer consists of two distinct bonded polymeric chains, in a configuration such as

\[ A-A-A-A-B-B-B-B. \]

Polymer chains have more coiled than stretched microstates, so entropic forces favour a coiled chain configuration, resulting in a disordered phase in a diblock copolymer melt at high temperature. An effective repulsion between monomers A and B counteracts this, preferring a stretched configuration with maximum separation between unlike monomers. These two effects compete, leading to the formation of various phases of ordered microdomains of A and B through a process known as self-assembly (see Figure 1). Observed structures of these microdomains include lamellar domains, hexagonally-packed cylindrical domains and body-centred cubic lattices of spherical domains. Self-assembly makes these materials attractive for applications where the creation and control of structures on nanometer scales is desirable – for example, as lithographic templates for nanowires and photonic crystals.

While perfectly periodic equilibrium phases are well-described by theory, the processing of DBCP melts requires an understanding of the dynamics of the formation of these mesophases.
An illustration of the location of an individual polymer within a melt in the lamellar and cylindrical mesophases, respectively.

For example, if one crosses a phase boundary while processing a block copolymer, a non-uniform phase can result due to the nucleation of a droplet of the stable structure out of the original (metastable) structure.

In addition, the equilibrium behaviour of DBCP melts and their dynamics are altered in the presence of a surface field, creating novel microstructures in confined spaces, including concentric lamellae, helical and gyroid structures. The dynamics of the formation of these microstructures is largely unexplored to date.

2. The Brazovskii Model for Diblock Copolymers

The behaviour of DBCP melts is found to be dependent on only two parameters \([1]\). The first, \(f\), is the quotient \(\frac{N_A}{N}\), where \(N_A\) is the number of \(A\) monomers in a polymer chain and \(N\) is the total number of monomers in the chain (the polymerization index). The second parameter is the product \(\chi N\), where \(\chi\) is the Flory-Huggins parameter, a temperature dependent parameter characterizing the monomer-monomer interaction strength.

We use the Landau-Brazovskii model \([2]\) to describe DBCP melts in the weak segregation limit. We define an order parameter \(\phi(r)\) to describe the thermal average of the deviation of the melt from homogeneity,

\[
\phi(r) = ((1 - f)\rho_A(r) - f\rho_B(r)),
\]

where \(\rho_A(r)\) and \(\rho_B(r)\) are the densities of monomers \(A\) and \(B\) respectively. In a high temperature, disordered phase, \(\phi(r)\) is zero everywhere, because the ratio of \(A\) to \(B\) monomers at any point in the melt is dependent only on \(f\). As the melt is cooled and ordered structure forms, \(\phi(r)\) deviates from zero and becomes periodic.

We encode the competing actions of the entropic force and the monomer-monomer interaction in the many-chain Edwards Hamiltonian \([3]\). Following the method of Leibler \([1]\) and Ohta and Kawasaki \([4]\), we use the tools of statistical mechanics to expand the free energy of the melt in powers of the order parameter. We use the de Gennes random phase approximation \([5]\) to write the expansion for interacting monomers in terms of the Green’s functions derived for non-interacting monomers moving in a mean field describing interactions.

We truncate this free energy expansion at fourth order and take the weak segregation limit by expanding around a single dominant scattering wavevector \(q_0\) in reciprocal space, corresponding to the periodicity of the melt. This approximation is valid in a regime close to the mean-field critical point. The resulting expression for the free energy in real space can be written in the form used by Wickham, Shi and Wang \([6]\):
\begin{equation}
F = \frac{1}{V} \int \frac{dr}{dV} \left\{ \frac{\xi^2}{2} (\nabla^2 + 1)\phi^2 + \frac{\tau}{2} \phi^2 - \frac{\gamma}{3!} \phi^3 + \frac{1}{4!} \phi^4 \right\}.
\tag{2.2}
\end{equation}

We have absorbed the dependence on the wavevector $q_0$ through a change of variables. Here $V$ is the volume of the melt, while $\xi$, $\tau$ and $\gamma$ are dependent on the parameters $f$ and $\chi N$. In fact, only $\tau$ is dependent on $\chi N$, and is thus the only variable dependent on the temperature of the melt. Also, $\gamma$ depends on the asymmetry of the copolymer composition, and is zero when $f = 0.5$.

3. Simulating the Dynamics of DBCP Melts

The code we have adapted to run on graphics processing units (GPUs) was originally developed by Robert Wickham’s group at the University of Guelph. To model the kinetics of a DBCP melt, we evolve the system in time governed by a diffusion equation,

\begin{equation}
\frac{\partial \phi}{\partial t} = -\nabla^2 F = -\nabla^2 \left\{ \xi^2 [(\nabla^4 + 2\nabla^2 + 1)\phi] + \tau \phi - \frac{\gamma}{2} \phi^2 + \frac{1}{6} \phi^3 \right\}.
\tag{3.1}
\end{equation}

The field thus evolves towards spatial configurations that minimize the free energy of the system.

To implement the kinetics numerically, equation (3.1) is integrated at every discretized timestep, using a real space finite difference approach. We initialize a field $\phi(x, y, z)$ in a mesh in three dimensions. We then either impose periodic boundary conditions to simulate an infinite melt in the study of nucleation, or impose a cylindrical mask on the melt in the $x$ and $y$ directions to model confinement.

We implement a second-order accurate Laplacian operator, calculated by averaging the gradient between each point on the mesh and its nearest neighbours,

\begin{equation}
\nabla^2 O \approx \frac{1}{6} (\Delta O_x + \Delta O_y + \Delta O_z),
\tag{3.2}
\end{equation}

where, for example

\begin{equation}
\Delta O_x = O(x-1, y, z) + O(x+1, y, z) - 2O(x, y, z)
\tag{3.3}
\end{equation}

and $O$ is any operand. This function is applied to $\phi$ to calculate $\nabla^2 \phi$, then again to $\nabla^2 \phi$ to calculate $\nabla^4 \phi$. Finally, the operator is applied to $\frac{DF}{\delta \phi}$ to calculate the right hand side of equation (3.1). Because each application of the Laplacian operator is dependent on nearest neighbours, the final right hand side of equation (3.1) is equivalent to the application of a third-to-nearest neighbour mask, in accordance with the prescriptions of a finite difference approach.

To model confinement we impose Neumann boundary conditions at the confining surface, setting those gradients that cross the boundary to zero. We then use a fourth order Runge-Kutta integration technique \cite{7} to integrate the right-hand side of equation (3.1).

Because the discretized Laplacian of the field at a point in the mesh is dependent only on the values of its nearest neighbours and all other operations performed on each point in the mesh are independent of neighbours, it is natural to turn to parallelization to evolve the field. The original code of the Wickham group used the Message Passing Interface (MPI) to achieve parallelization by dividing the melt into slices along the $z$-axis, and computing each slice on a different processor, with MPI calls passing boundary values between processors.

3
While this is an improvement over serially iterating over every value of the field, the fact that the evolution of each mesh point depends only on its nearest neighbours makes much broader parallelization possible. We turn to graphics processing units (GPUs) and the NVIDIA CUDA architecture to achieve this parallelization.

4. Implementation on Graphics Processing Units

4.1. The CUDA Architecture

Unlike traditional Central Processing Units (CPUs), Graphics Processing Units (GPUs) devote most of their architecture to computation, with massive numbers of arithmetic logic units (ALUs) and less space devoted to caching and flow control operations [8]. These many ALUs allow many units of computation to be executed in parallel. The NVIDIA® CUDA™ architecture defines an instruction set that takes advantage of this parallelization, by allowing the programmer to replace serial loop structures with kernels, which represent an operation that can be performed in parallel over an entire data set. These kernels are executed as threads, arranged into warps run as a single computational entity on multiple ALUs.

To take advantage of this parallelization, we identified those sections of the original code where an operation was performed for each element in the data set, characterised by a 3-dimensional for-loop. We then wrote kernels to replace those sections of the code, operating on an initial field configuration transferred from traditional memory to GPU device memory at the beginning of the computation.

We left the MPI structure of the code intact, creating a hybrid CUDA-MPI code. Some latency is introduced in transferring data from the GPU to be sent to neighbouring processors in MPI codes, but the ability to parallelize both within GPUs and across many cores outweighs this deficiency.

4.2. Implementation of the Laplacian

Threads are arranged into blocks. Each block has an area of shared memory associated with it, which can access global device memory in parallel in a process known as coalesced memory access. In cases where the same data is accessed by many threads, shared memory can provide impressive performance improvements. We explored this approach in tandem with a global memory approach for the implementation of the Laplacian.

Following work done by Leist, Playne and Hawick [9], who investigated a similar system, we implemented two kernels for the Laplacian operator. The first accessed data for nearest neighbours from global memory, while the second took advantage of shared memory and coalesced memory access to call nearest neighbour information without making multiple calls to global device memory. A third kernel, using texture memory, was discarded in the early stages of development as too complex to implement practically in concurrence with the original MPI structure of the code. Leist, Playne and Hawick [9] implement a texture memory kernel in two dimensions and present results indicating that investigation into texture memory may produce further performance improvements, but we leave this for future work.

One subtlety involved with coalesced memory access is the treatment of border threads. A block can hold a maximum of 512 threads [8], indexed in any number of dimensions. In order to take advantage of CUDA’s parallelization, kernels must be free of branches, where threads in a warp take different execution paths such as through an if-statement. Thus, border threads cannot be treated differently from internal threads. As such, we follow Leist, Playne and Hawick in implementing non-updating border threads, which hold values to allow their nearest neighbours inside the block to update without a kernel branch.

In both cases, we used the maximum block size of 512 in order to exploit GPU parallelization to its fullest extent. When reading from global memory, where the position of adjacent threads is less important, we mapped to a $16 \times 32$ block of threads. When reading from shared

---

High Performance Computing Symposium (HPCS2010) IOP Publishing
Journal of Physics: Conference Series 256 (2010) 012008
doi:10.1088/1742-6596/256/1/012008
memory, we initialized in three dimensions to a $16 \times 8 \times 4$ block size, including the non-updating border threads. The latter specification was implemented to ensure that one dimension of the block was of a size sufficient to take advantage of all 16 consecutive threads that may be read simultaneously.

The authors of Ref. [9] consider a two-dimensional system ringed by a one-dimensional square of border threads. We consider a three-dimensional system and a two-dimensional surface of border threads, resulting in further latency caused by the extra threads introduced. This problem could be alleviated through further work, following methods documented in Ref. [10]$,^1$ where a two-dimensional tile is used as the basic block.

5. Performance Results
We benchmarked the performance of the old MPI code and compared it with results from the GPU code. The performance improvements are shown in Figure 5. We can see from Figure 5 that at mesh dimension $N = 384$, where $N = 3 \sqrt[3]{L_x L_y L_z}$ and $L_k$ is the number of elements in the mesh in the $k$-dimension, the performance improvement is a factor of 30, a result that facilitates much of the other work presented in this paper.

![Figure 2. Linear and logarithmic plots of the performance gains of the various GPU kernels, compared against the CPU implementation. Both implementations use MPI parallelization on 8 cores. Mesh dimension $N = 3 \sqrt[3]{L_x L_y L_z}$.](image)

$^1$ We acknowledge the referees of this paper for their help in directing us to this reference.
Leist, Playne and Hawick’s studies in two dimensions found that a kernel using shared memory was faster than one using global memory for any value of $N^3$ greater than approximately $2 \times 10^4$ [9]. However, our global memory kernel was more efficient than the shared memory kernel for any $N^3$ less than $2 \times 10^7$ (see Figure 3). We suspect that this discrepancy is due to the three-dimensional nature of our problem requiring many more non-updating border threads and thus creating more latency in the code.

6. Droplet Nucleation

The performance improvements discussed in the previous section are important when a large system size is required. A situation where this is the case is nucleation. In a first-order phase transition such as the lamellar-cylinder phase transition in DBCP melts, there is a region between the phase boundary and the spinodal line where the phase is metastable (see Figure 4). If fluctuations in the field nucleate a sufficiently large droplet of the stable phase, characterised by critical size, the droplet will overcome the energy barrier preventing the phase transition and grow, resulting in the entire system transitioning to the stable phase. Both disorder-order transitions [11, 12] and order-order transitions [13, 6] have been studied theoretically. Our research focuses on order-order transitions.

We manually nucleate a droplet of the stable cylindrical phase in a lamellar melt initialized at a temperature where it is metastable (see Figure 5). Very close to the coexistence boundary and away from the spinodal regime, the size of the critical droplet, which will grow and effect the phase transition, is much larger than the characteristic length associated with the critical wavevector $\frac{2\pi}{q_0}$. These two differing length scales result in the need for a large system size to model nucleation accurately.

Figure 6 shows both results obtained using MPI [14] and new results obtained using the MPI-GPU hybrid code. The hybrid code demonstrably tests nucleation theory in regions much closer to coexistence than previously investigated. The size of the error bars on the new data can be shrunk by further work with the code, as they represent the region in which it may not be clear after $10^6$ timesteps whether the droplet has grown or shrunk.
Figure 4. The phase diagram for the lamellar-cylinder transition as derived by Wickham, Shi and Wang [6]. Solid lines - phase boundaries; dashed line - disorder-order spinodal; dotted line: lamellar-to-cylinder spinodal; dashed-dotted line: cylinder-to-lamellar spinodal. Other phase boundaries, such as for the bcc and gyroid phases, are not shown. The central point is the mean-field critical point, where a symmetric DBCP will undergo a second-order phase transition.

Figure 5. A cross-section of a cylindrical bulk where a lamellar droplet has just been nucleated in our simulation. In this case $R \approx 6L_0$. 
Figure 6. Half-length of major axis of critical droplet, measured in cylinder spacings, as a function of $\tau$.

7. Dynamics in a Cylindrical Nanopore

If we place a diblock copolymer melt into contact with a surface, it is possible that the substance of which the surface is composed may interact unequally with monomers $A$ and $B$. This would favour configurations of the system where one species of monomer is closer to the surface. The combination of confinement and the presence or absence of a surface field produces novel structures, such as concentric lamellar structures, helices and double helices, particularly at small pore radius.

The equilibrium behaviour of these morphologies has been derived in detail [15, 16, 17, 18, 19, 20], and is dependent on block asymmetry (the value of $f$), surface field strength and the incommensurability of the diameter of the confined space with the natural periodicity of structures formed in the bulk. Pore diameters $D_p = (n + 0.5)L_0$, where $n$ is an integer and $L_0$ is the bulk structure period, are expected to deviate most starkly from bulk structures. In addition, narrow pores with high curvature are expected to create large deviations from bulk behaviour.

The dynamics of the formation of these structures has been largely unexplored. Many of the most interesting novel structures, such as helical structures, appear at very small pore radius. A coarse mesh at this small system size introduces unwanted effects, so a very fine mesh is required. To maintain the stability of the code we thus used a very short timestep, which would have made simulation intractable without the performance improvements of the GPU code. We use the new code to study two features of interest - the growth of structure from a disordered phase, and the formation of chiral structures such as helical structures from an achiral phase.

We use the same form of the surface field used in Ref. [21]. If $R_p$ is the radius of the pore, $\lambda$ is the decay length of the interaction and $\sigma$ is the interaction cutoff length, then the interaction strength $H(r)$ is defined to be

$$H(r) = \begin{cases} H_S \frac{\exp\{(\sigma + |r| - R_p)/\lambda\} - 1}{\exp(\sigma/\lambda) - 1} & R_p - \sigma \leq |r| \leq R_p \\ 0 & \text{otherwise.} \end{cases}$$

(7.1)
We follow Li and Wickham in setting $\sigma = 0.5R_g$ and $\lambda = 0.25R_g$, where $R_g$ is the radius of gyration of the copolymer. This prescription sets the strength of the monomer-pore interaction to be of roughly the same order as the monomer-monomer interaction strength.

7.1. The Growth of Structure

We first set the surface field strength $H_s = 0.05$, and the pore diameter to be larger than the characteristic period of lamellae in the bulk $L_0 = \frac{2\pi}{\theta}$. We observe that the structure of concentric lamellae grows from the pore wall inwards, as shown in Figure 7, with outer rings showing much stronger segregation than inner rings. This is important in nanotechnology applications where many concentric rings are desired, as each ring appears to take longer to form than the previous one. After $10^5$ timesteps, the largest pore radius we simulated, at $\frac{R_p}{L_0} = 5.1$ ($R_p$ the pore radius), is still not completely formed (See Figure 7c).

![Figure 7](image)

**Figure 7.** The growth of concentric lamellar structure for $H_s = 0.05$, $f = 0.45$, $\chi N = 15$, $\frac{R_p}{L_0} = 5.1$. Stronger segregation is indicated towards the red end of the spectrum, while weaker segregated areas are shown in blue. Timesteps are $t=0$; $t=1000$; $t=50000$.

7.2. Chirality

We next chose a finer mesh and examined structures found when $1 < \frac{D_p}{L_0} < 3$ ($D_p$ the pore diameter). Many of these structures are chiral, and we were particularly interested in the transition from a uniform, disordered phase to a chiral phase. The route to chiral structure is in this case through a gyroid-like phase, which then transforms asymmetrically into linked disc structures, which in turn accrete into the helical phases observed in the final states: a double helix in the case of $\frac{D_p}{L_0} = 2.4$; a single helix for $\frac{D_p}{L_0} = 1.6$ (see Figure 8); and interestingly more slowly forming disc-like structures at other radii, which we expect at long times to form one of the equilibrium structures predicted by Li and Wickham [19].

These results are particularly interesting because the theoretical bulk prediction at this temperature is of lamellar structure, while helical structures are generally thought to appear through the distortion of a cylindrical phase by the presence of the nanopore [19, 21]. This may be explained by a stress placed on the system by the imposition of periodic boundary conditions along the $z$-axis, which may stabilize the helical phase.

To explore chiral structure in more depth, we increased the asymmetry of the polymers in the melt by decreasing $f$, and initialized in an achiral bcc sphere phase, evolving in the presence of a nanopore at a temperature where a cylindrical phase is predicted in the bulk. The results of this simulation are shown in Figure 9. It can be seen that the bcc nearest neighbours mark
Figure 8. The growth of chiral structure for $H_s = 0.05$, $f = 0.45$, $\chi N = 15$, $D_p L_0 = 1.6$, $t = 0$; $1 \times 10^5$; $2 \times 10^5$; $1 \times 10^6$

out the final helical structure. Chirality appears to be chosen at random, with each handedness equally likely to be found over multiple runs.

Figure 9. The growth of helical structure from an initial bcc sphere phase. $H_s = 0.05$, $f = 0.4$, $\chi N = 11.8$, $D_p L_0 = 1.6$, $t = 0$; $5 \times 10^4$; $1 \times 10^5$; $2 \times 10^5$.

8. Conclusion

In summary, we have demonstrated the applicability of graphics processing units (GPUs) to polymer dynamic field theory, finding a factor of 30 performance improvement over an existing MPI code at large system size. We have used this performance improvement to study two aspects of the dynamics of diblock copolymer melts in regimes that were previously computationally intractable: droplet nucleation close to phase coexistence and the dynamics of structure growth and chirality in the presence of a cylindrical nanopore.
Future work on the simulation code itself should focus on reducing latency due to the passing of boundary information from one GPU to another, which is currently a hurdle in the performance of the code when run on a large number of GPUs. This would be achieved by implementing non-updating border threads on each GPU core.

In addition, the performance of the code would allow for a full study of the dynamics of DBCP melts under nanopore confinement across regimes of temperature, asymmetry and pore radius.

This work was supported by the Perimeter Scholars International Program, and by NSERC.

References

[1] Leibler L 1980 Macromolecules 13 1602–1617
[2] Brazovskii S 1975 Sov. Phys. JETP 41 6725–6760
[3] Edwards S 1966 Proc. Phys. Soc. (London) 88 265
[4] Ohta T and Kawasaki K 1986 Macromolecules 19 2621–2632
[5] de Gennes P 1970 J. Phys. (Paris) 31 235
[6] Wickham R, Shi A and Wang Z 2003 J. Chem. Phys. 118 10293–10305
[7] Butcher J 2003 Numerical methods for ordinary differential equations (John Wiley and Sons)
[8] NVIDIA® Corporation CUDA™ 2.0 Programming Guide Available at: http://www.nvidia.com/ [last accessed April 2010]
[9] Leist A, Playne D and Hawick K 2009 Concurrency Computat.: Pract. Exper. 21 2400–2437
[10] Micikevicius P 3d finite difference computation on gpus using cuda Available at: http://developer.download.nvidia.com/CUDA/CUDA_Zone/papers/gpu_3d_fd_rev.pdf [last accessed August 2010]
[11] Fredrickson G and Binder K 1989 J. Chem. Phys. 91 7265–7275
[12] Hohenberg P and Swift J 1995 Phys. Rev. E 52 1828–1845
[13] Nonomura M and Ohta T 2001 J. Phys.: Condens. Matter 13 9089–9112
[14] Spencer R and Wickham R 2010 Unpublished
[15] He Z, Song M, liang H and Pan C 2001 J. Chem. Phys. 114 10510–10513
[16] Sevinik G, Zvelindovsky A, Fraaije J and Huinink H 2001 J. Chem. Phys. 115 8226–8230
[17] Y Wu et al 2004 Nature Materials 3 816–822
[18] Feng J, Liu H and Hu Y 2006 Macromol. Theory Simul. 15 675–685
[19] Li W and Wickham R 2006 Macromolecules 39 8492–8498
[20] Ma M, Thomas E, Rutledge G, Yu B, Li B, Jin Q, Ding D and Shi A C 2010 Macromolecules 43 3061–3071
[21] Li W and Wickham R 2009 Macromolecules 42 7530–7536