Stochastic Event-Driven Molecular Dynamics

Aleksandar Donev,[1] Alejandro L. Garcia,[2] and Berni J. Alder[1]

[1]*Lawrence Livermore National Laboratory,*

P.O.Box 808, Livermore, CA 94551-9900

[2]*Department of Physics, San Jose State University, San Jose, California, 95192*

Abstract

A novel Stochastic Event-Driven Molecular Dynamics (SEDMD) algorithm is developed for the simulation of polymer chains suspended in a solvent. SEDMD combines Event-Driven Molecular Dynamics (EDMD) with the Direct Simulation Monte Carlo (DSMC) method. The polymers are represented as chains of hard spheres tethered by square wells and interact with the solvent particles with hard core potentials. The algorithm uses EDMD for the simulation of the polymer chain and the interactions between the chain beads and the surrounding solvent particles. The interactions between the solvent particles themselves are not treated deterministically as in EDMD, rather, the momentum and energy exchange in the solvent is determined stochastically using DSMC. The coupling between the solvent and the solute is consistently represented at the particle level retaining hydrodynamic interactions and thermodynamic fluctuations. However, unlike full MD simulations of both the solvent and the solute, in SEDMD the spatial structure of the solvent is ignored. The SEDMD algorithm is described in detail and applied to the study of the dynamics of a polymer chain tethered to a hard wall subjected to uniform shear. SEDMD closely reproduces results obtained using traditional EDMD simulations with two orders of magnitude greater efficiency. Results question the existence of periodic (cycling) motion of the polymer chain.

I. INTRODUCTION

Driven by nanoscience interests, it has become necessary to develop tools for hydrodynamic calculations at the atomistic scale [1–5]. Of particular interest is the modeling of polymers in a flowing "good" solvent for both biological (e.g., cell membranes) and engineering (e.g., micro-channel DNA arrays) applications [4, 6]. The most widely studied polymer models are simple linear bead-spring; freely-jointed rods; or worm-like chains. Such models have been parameterized for important biological and synthetic polymers. Much theoretical, computational, and experimental knowledge about the behavior of these models has been accumulated for various representations of the solvent. However, the multi-scale nature of the problem for both time and length is still a challenge for simulations of reasonably large systems over reasonably long times. Furthermore, the omission in these models of the *explicit* coupling between the solvent and the polymer chain(s) requires the introduction of adjustable parameters (e.g., friction coefficients) to be determined empirically. The algorithm presented here overcomes this deficiency for a linear polymer chain tethered to a hard wall and subjected to a simple linear shear flow [7–11]. Of particular interest is the long-time dynamics of the polymer chain [7, 9, 10, 12, 13] and any effects of the polymer motion on the flow field.

Brownian dynamics is one of the standard methods for coupling the polymer chains to the solvent [14, 15]. The solvent is only implicitly represented by a coupling between the polymer beads and the solvent in the form of stochastic (white-noise) forcing and linear frictional damping. The flow in the solvent is not explicitly simulated, but approximated as a small perturbation based on the Oseen tensor. This approximation is only accurate at large separations of the beads and at sufficiently small Reynold's numbers. Even algorithms that do model the solvent explicitly via Lattice Boltzmann (LB) [16], incompressible (low Reynolds number) CFD solvers [17–19], or multiparticle collision dynamics [20–23], typically involve phenomenological coupling between the polymer chain and the flowing fluid in the form of a linear friction term based on an effective viscosity. Furthermore, solvent fluctuations in the force on the polymer beads are often approximated without fully accounting for spatial and temporal correlations. Finally, the reverse coupling of the effect of the bead motion on the fluid flow is either neglected or approximated with delta function forcing terms in the continuum fluid solver [24]. More fundamentally, continuum descriptions of flow at micro

and nanoscales are known to have important deficiencies [1, 3] and therefore it is important to develop an all-particle algorithm that is able to reach the long times necessary for quantitative evaluation of approximate, but faster, algorithms.

The most detailed (and expensive) modeling of polymers in flow is explicit molecular dynamics (MD) simulation of both the polymer (solute) and the surrounding solvent [11, 25]. Multi-scale algorithms have been developed to couple the MD simulation to Navier-Stokes-based computational fluid dynamics (CFD) calculations of the flow field [8]. However, the calculation time still remains limited by the slow molecular dynamics component. Thus the computational effort is wasted on simulating the structure and dynamics of the solvent particles, even though real interest lies in the polymer structure and dynamics, and their coupling to the fluid flow. Our algorithm replaces the deterministic treatment of the solvent-solvent interactions with a stochastic momentum exchange operation, thus significantly lowering the computational cost of the algorithm, while preserving microscopic details in the solvent-solute coupling.

Fluctuations drive the polymer motion and must be accurately represented in any model. Considerable effort has been invested in recent years in including fluctuations directly into the Navier-Stokes (NS) equations and the associated CFD solvers [5, 17, 26]. Such fluctuating hydrodynamics has been coupled to molecular dynamics simulations of polymer chains [19], but with empirical coupling between the beads and the fluid as discussed above. To avoid the empirical coupling, the solvent region could be enlarged by embedding the atomistic simulations of the region around the polymer chain in a fluctuating hydrodynamics region. The bidirectional coupling between the continuum and particle regions has to be constructed with great care so that both fluxes and fluctuations are preserved [27]. A well-known problem with such multiscale approaches is that the finest scale (atomistic simulation) can take up the majority of computational time and thus slow down the whole simulation. By using DSMC the cost of the particle region can be made comparable to that of the continuum component.

The Stochastic Event-Driven Molecular Dynamics (SEDMD) algorithm presented here combines Event-Driven Molecular Dynamics (EDMD) for the polymer particles with Direct Simulation Monte Carlo (DSMC) for the solvent particles. Note that our algorithm is similar to a recent algorithm developed for soft interaction potentials combining time-driven MD with multiparticle collision dynamics [23]. In SEDMD, the polymers are represented as

chains of hard spheres tethered by square wells. The solvent particles are realistically smaller than the beads and are considered as hard spheres that interact with the polymer beads with the usual hard-core repulsion. The algorithm processes true (deterministic, exact) binary collisions between the solvent particles and the beads, without any approximate coupling or stochastic forcing. However, the solvent particles themselves do not directly interact with each other, that is, they can freely pass through each other as for an ideal gas. Deterministic collisions between the solvent particles are replaced with momentum- and energy-conserving stochastic DSMC collisions between nearby solvent particles. This gives realistic hydrodynamic behavior of the solvent similar to that of a true hard-sphere liquid. However, the DSMC fluid cannot directly be compared to an EDMD fluid because the two fluids are different. Notably, the internal structure and the associated non-ideal equation of state (EOS) of the hard-sphere liquid is lost when using DSMC. We are currently developing DSMC variants that produce more realistic fluid behavior, in particular, a non-trivial structure factor and a thermodynamically-consistent EOS.

The fundamental ideas behind our algorithm are described next, and further details are given in Section III. Section V gives results from the application of the algorithm to the tethered polymer problem, and some concluding remarks are given in Section VI.

II. HYBRID COMPONENTS

In this section we briefly describe the two components of the SEDMD algorithm: The stochastic handling of the solvent and the deterministic handling of the solute particles. These two components are integrated (i.e., *tightly* coupled) into a single event-driven algorithm in Section III.

A. Solvent DSMC Model

The validity of the incompressible Navier-Stokes continuum equations for modeling microscopic flows has been well established down to length scales of $10 - 100nm$ [3]. However, there are several issues present in microscopic flows that are difficult to account for in models relying on a purely PDE approximation. Firstly, it is not *a priori* obvious how to treat boundaries and interfaces well, so as account for the non-trivial (possibly non-linear) cou-

pling between the flow and the microgeometry. Furthermore, fluctuations are not typically considered in Navier-Stokes solvers, and they can be very important at instabilities [28] or in driving polymer dynamics. Finally, since the grid cell sizes needed to resolve complex microscopic flows are small, a large computational effort (comparable to DSMC) is needed even for continuum solvers. An alternative is to use particle-based methods, which are explicit and unconditionally stable and rather simple to implement. The solvent particles are directly coupled to the microgeometry, for example, they directly interact with the beads of a polymer. Fluctuations occur naturally with the correct spatio-temporal correlations.

Several particle methods have been described in the literature, such as MD [25], dissipative particle dynamics (DPD) [29], and multi-particle collision dynamics (MPCD) [2, 23]. Our method is similar to MPCD (also called stochastic rotation dynamics or the Malevanets-Kapral method), and in fact, both are closely related to the Direct Simulation Monte Carlo (DSMC) algorithm of Bird [30]. The key idea behind DSMC is to replace deterministic interactions between the particles with stochastic momentum exchange (collisions) between nearby particles. Time-driven (traditional) DSMC involves the following steps:

Advection Every particle i is propagated ballistically by a fixed time step Δt, $\mathbf{r}_i \leftarrow \mathbf{r}_i + \Delta t \mathbf{v}_i$.

Sorting Particles are sorted into cells, each containing a few (e.g., 2-10) particles.

Collision In each cell, a certain number $N_{coll} = \Gamma_c \Delta t$ of random pairs of particles are chosen to undergo energy- and momentum-conserving stochastic collisions. The collision rate Γ_c and the pairwise probability distributions are chosen based on kinetic theory.

Formally, DSMC can be seen as a method for solving the Boltzmann transport equation for a low-density gas, however, it is not limited to gas flows [31–33]. Our purpose for using DSMC is as a replacement for expensive MD, preserving the essential hydrodynamic "solvent" properties: local momentum conservation, and linear momentum exchange on length scales comparable to the particle size, and a similar fluctuation spectrum.

In the multiparticle collision variant of this algorithm originally proposed by Kapral, the traditional DSMC collection of binary collisions is replaced by a multi-particle collision in which the velocities of all particles in the cell are rotated by a random amount around the average velocity [2, 23]. This change improves efficiency but at the cost of some artificial

effects such as loss of Galilean invariance. These problems can be corrected and the method has been successfully used in modeling polymers in flow by including the beads, considered as (massive) point particles, in the stochastic momentum exchange step [20, 22, 34].

It is important to note that the DSMC fluid is not a true hard-sphere liquid, as would be simulated by using full MD. Notably, the DSMC fluid has no internal structure and has an ideal gas equation of state (EOS), and is thus very compressible. For subsonic flows this compressibility does not qualitatively affect the results as the DSMC fluid will behave similarly to an incompressible liquid, however, the (Poisson) density fluctuations in DSMC are significantly larger than those in realistic liquids. Furthermore, the speed of sound is small (comparable to the average speed of the particles) and thus subsonic (Mach number less than one) flows are limited to relatively small Reynolds numbers [1]. The Consistent Boltzmann Algorithm (CBA) [32, 33], as well as algorithms based on the Enskog equation [35, 36], have demonstrated that DSMC fluids can have dense-fluid compressibility. A similar algorithm was recently constructed for MPCD [37]. We are currently developing several thermodynamically-consistent variants of DSMC that give fluids with a non-ideal EOS and will report our findings in future work.

B. Polymer MD Model

Polymer chains in a solvent are often modeled using continuous pair potentials and time-driven MD (TDMD), in which particles are synchronously propagated using a time step Δt, integrating the equations of motion along the way. Typically the polymer is assumed to be in a good solvent, that is, that the effective interaction between polymer beads is repulsive and thus the polymer chain does not collapse to a globule but is extended. The polymer beads are represented as spherical particles that interact with other beads and solvent particles with short-range repulsive pair potentials, such as the positive part of the Lennard-Jones potential. Additionally, beads are connected via finitely-extensible FENE or worm-like springs in order to mimic chain connectivity and elasticity [25]. Finally, stochastic

[1] For a low-density gas the Reynolds number is $Re = M/K$, where $M = v_{flow}/c$ is the Mach number, and the Knudsen number $K = \lambda/L$ is the ratio between the mean free path λ and the typical obstacle length L. This shows that subsonic flows can only achieve high Re flows for small Knudsen numbers, i.e., large numbers of DSMC particles.

forces may be present to represent the solvent.

The time steps required for integration of the equations of motion in the presence of the strongly repulsive forces is small and TDMD cannot reach long time scales even after parallelization. An alternative is to use hard spheres instead of soft particles, allowing replacement of the FENE springs with square-well tethers, thus avoiding the costly force evaluations in traditional MD. Hard sphere MD is most efficiently performed using event-driven molecular dynamics (EDMD) [38–41]. If the detailed structure and energetics of the liquid is not crucial, such EDMD algorithms can be just as effective as TDMD ones but considerably faster. The essential difference between EDMD and TDMD is that EDMD is asynchronous and there is no time step, instead, collisions between hard particles are explicitly predicted and processed at their exact (to numerical precision) time of occurrence. Since particles move along simple trajectories (straight lines) between collisions, the algorithm does not waste any time simulating motion in between events (collisions).

Hard-sphere models of polymer chains have been used in EDMD simulations for some time [40, 42, 43]. These models typically involve, in addition to the usual hard-core exclusion, additional *square well* interactions to model chain connectivity. The original work by Alder *et al.* on EDMD developed the collisional rules needed to handle arbitrary square wells [38]. Infinitely high wells can model tethers between beads, and the tethers can be allowed to be broken by making the square wells of finite height, modeling soft short-range attractions. Recent studies have used square well attraction to model the effect of solvent quality [41]. Even more complex square well models have been developed for polymers with chemical structure and it has been demonstrated that such models, despite their apparent simplicity, can successfully reproduce the complex packing structures found in polymer aggregation [42, 43]. Recent work on coupling a Kramer bead-rod polymer to a Navier Stokes solver has found that using hard rods instead of soft interactions not only rigorously prevents rod-rod crossing, but also achieves a larger time step, comparable to the time step of the continuum solver [21].

This study is focused on the simplest model of a polymer chain, namely, a linear chain of N_b particles tethered by unbreakable bonds. This is similar to the commonly-used freely jointed bead-spring FENE model model used in time-driven MD. The length of the tethers

has been chosen to be $1.1D_b$, where D_b is the diameter of the beads [2]. The implementation of square-well potentials is based on the use of near-neighbor lists (NNLs) in EDMD, and allows for the specification of square-well interactions for arbitrary pairs of near neighbors. In particular, one can specify a minimal $L_t^{min} \geq D_b$ and maximal distance $L_t^{max} > L_t^{min}$ for arbitrary pairs of near neighbors. Here the maximal distance L_t^{max} represents the tether length between neighboring beads. A value $L_t^{min} > D_b$ can be used to emulate chain rigidity (i.e., a finite persistence length) by using second nearest-neighbor interactions between chain beads.

III. DETAILS OF HYBRID ALGORITHM

In this section the hybrid EDMD/DSMC algorithm, which we name Stochastic EDMD (SEDMD), is described in detail. Only a brief review of the basic features of EDMD is given and the focus is on the DSMC component of the algorithm and the associated changes to the EDMD algorithm described in detail in Ref. [39]. A more general description of asynchronous event-driven particle algorithms is given in Ref. [44].

Asynchronous event-driven (AED) algorithms process a sequence of *events* (e.g., collisions) in order of increasing event time t_e. The time of occurrence of events is predicted and the event is scheduled to occur by placing it an *event queue*. The simulation iteratively processes the event at the head of the event queue, possibly scheduling new events or invalidating old events. One *impending event* per particle i, $1 \leq i \leq N$, is scheduled to occur at time t_e with partner p (e.g., another particle j). The particle position $\mathbf{r}_i$ and velocity $\mathbf{v}_i$ are *only* updated when an event involving particle i is processed and the time of last update t_i is recorded. We will refer to this procedure as a *particle update*.

We note that traditional *synchronous time-driven* (STD) algorithms with a *time step* Δt are a trivial variant of the more general AED class. In particular, in an STD algorithm events occur at equispaced times and each event is a *time step* requiring an update of all of the particles. Our SEDMD algorithm processes a mixture of events involving single particles or pairs of particles with time steps that involve the simultaneous (synchronous) update of a large collection of particles.

[2] Note that the hard-sphere model rigorously prevents chain crossing if the tether length is less than $2D_b$ in two dimensions and $\sqrt{2}D_b$ in three dimensions.

Every particle i belongs to a certain specie s_i. We focus on a system in which a large fraction of the particles belong to a special specie s_{DSMC} representing DSMC particles (e.g., solvent molecules). These DSMC particles do not interact with each other via hard-core repulsion, but they do interact with particles of other species. We focus on the case when the non-DSMC particles are localized in a fraction of the simulation volume, while the rest of the volume is filled with DSMC particles. This will enable us to treat the majority of DSMC particles sufficiently far away from non-DSMC particles more efficiently than those that may collide with non-DSMC particles.

Before describing the SEDMD algorithm in detail, we discuss the important issue of efficiently searching for nearby pairs of particles.

A. Near Neighbor Searches

When predicting the impending event of a given particle i, the time of potential collision between the particle and each of its nearby *neighbor particles* is computed [39, 44]. The DSMC algorithm also requires defining neighbor particles, that is, particles that may collide stochastically during the DSMC collision step. For efficiency, geometric techniques are needed to make the number of neighbors of a given particle $O(1)$ instead of $O(N)$.

In SEDMD we use the so-called *linked list cell* (LLC) method for neighbor searching in both the EDMD and DSMC components. The simulation domain is partitioned into N_{cells} cells as close to cubical as possible. Each particle i stores the cell c_i to which its centroid belongs, and each cell c stores a list $\mathcal{L}_c$ of all the particles it contains, as well as the total number of particles N_c in the cell. For a given interaction range, neighbors are found by traversing the lists of as many neighboring cells as necessary to ensure that all particles within that interaction range are covered. In traditional DSMC, only particles within the same cell are considered neighbors and thus candidates for collision. There are also variants of DSMC in which particles in nearby cells are included in order to achieve a non-ideal equation of state [35, 36]. Such variants can be used in SEDMD without any changes to the EDMD component of the algorithm.

1. Cell Bitmasks

In addition to the list of particles $\mathcal{L}_c$, each cell c stores a *bitmask* $\mathcal{M}_c$ consisting of $N_{\text{bits}} > N_s + 4$ bits (bitfields). These bits may be one (set) or zero (not set) to indicate certain properties of the cell, specifically, what species of particles the cell contains, whether the cell is event or time driven, and to specify boundary conditions. In order to distinguish the cells that contain non-DSMC particles from those that contain only DSMC particles, bit γ is set if the cell may contain a particle of specie γ. The bit is set whenever a particle of specie γ is added to the cell, and all of the masks are reset and then re-built (refreshed) periodically. When performing a neighbor search for a particle i, cells not containing particles of species that interact with specie s_i are easily found by OR'ing the cell masks with a specie mask, and are simply skipped. This speeds up the processing of DSMC particles since cells containing only DSMC particles will be skipped without traversing their lists of particles.

In SEDMD we will also need to distinguish those cells that are nearby non-DSMC particles, that is, that contain particles within the interaction range of some non-DSMC particle. Such cells will be treated using a fully event-driven (ED) scheme, while the remaining cells will be treated using a time-driven or mixed approach. We use one of the bits in the bitmasks, bit γ_{ED}, to mark *event-driven (ED) cells*. Specifically, bit γ_{ED} is set for a given cell whenever the cell is traversed during a neighbor search for a non-DSMC particle. This scheme correctly masks the cells by only modifying the neighbor search routine (iterator) without changing the rest of the algorithm, at the expense of a small overhead. We also mark the cells near hard-wall boundaries as ED cells. Cell bitmasks should be cleared and rebuilt periodically so as to prevent the fraction of ED cells from increasing. As will be seen shortly, it is necessary to introduce at least one "sticky" bit γ_{st} that has memory and is not cleared when cell bitmasks are refreshed. This bit will mark *unfilled cells*, as explained in Section III C.

2. Near Neighbor Lists

The cell size should be tailored to the DSMC portion of the algorithm and can become much smaller than the size of some of the non-DSMC particles. The LLC method becomes inefficient when the interaction range becomes significantly larger than the cell size because many cells need to be traversed. In this case the LLC method can be augmented with

the *near-neighbor list* (NNL) method, and in particular, the bounding sphere complexes (BSCs) method, as described in detail for nonspherical hard particles in Ref. [39]. We have implemented the necessary changes to the algorithm to allow the use of NNLs and BSCs (in addition to LLCs), and we used NNLs in our simulations of polymer chains in solution. The use of BSCs is not necessary for efficient simulations of polymer solutions if the size of the polymer bead is comparable to the size of the cells, which is the case for the simulations we report. We do not describe the changes to the algorithm in detail; rather, we only briefly mention the essential modifications.

For the purposes of DSMC it is important to maintain accurate particle lists $\mathcal{L}_c$ for all cells c, so that it is known which particles are in the same cell at any point in time. Therefore, transfers of particles between cells need to be predicted and processed even though this is not done in the NNL algorithm described in Ref. [39]. Near-neighbor lists are *only* built and maintained for DSMC particles that are in event-driven cells, essentially exactly as described in Ref. [39]. For a DSMC particle i that is not in an ED cell c_i we consider the smallest sphere enclosing cell c_i to be the (bounding) neighborhood (see Ref. [44]) of particle i and *only* update the (position of the) neighborhood when the particle moves to another cell. This ensures that neighbor searches using the NNLs are still exact without the overhead of predicting and processing *NNL update* events for the majority of the DSMC particles.

B. The SEDMD Algorithm

We have developed an algorithm that combines time-driven DSMC with event-driven MD by splitting the particles between ED particles and TD particles. Roughly speaking, only the particles inside event-driven cells are treated asynchronously as in EDMD. The rest of the particles are DSMC particles that are not even inserted into the event queue. Instead, they are handled using a time-driven (TD) algorithm very similar to that used in traditional DSMC.

It is important to note that the division of the DSMC particles between ED and TD handling is dynamic and does not necessarily correspond to the partitioning of the cells into ED and TD cells. As non-DSMC particles move, time-driven cells may be masked as event-driven. This does not immediately make the DSMC particles in such cells event-driven. Rather, time-driven DSMC particles are moved into the event queue only when a collision

with a non-DSMC particle is scheduled for them, when they move into a TD cell following a time step, or when restarting the event handling. Event-driven particles are removed from the event queue when they undergo cell transfer events into time-driven cells.

It is also possible to implement DSMC as a fully asynchronous event-driven (AED) algorithm and thus avoid the introduction of an external time scale through the time step Δt. The fully asynchronous algorithm introduces a novel type of event we term *stochastic* (DSMC) *collisions*, and it is discussed in more detail in Appendix A. Asynchronous processing has a few advantages over the traditional (synchronous) time-driven approach, notably, no errors due to time discretization [45] and improved efficiency at low collision rates. For high densities the collision rate is high enough that the computational cost is dominated by collision processing, and the asynchronous algorithm is actually less efficient due to the overhead of event queue operations. Additionally, time-driven handling has certain important advantages in addition to its simplicity, notably, the synchrony of the DSMC portion of the algorithm allows for parallelization and easy incorporation of algorithmic alternatives such as multi-particle or multi-cell collisions, adaptive open boundary conditions, etc.

1. Event Handling

The SEDMD algorithm handles events in order of increasing time of occurence, just as in EDMD. The main types of events in the SEDMD algorithm are:

Update Move particle i to the current simulation time t if $t_i < t$.

Transfer Move particle i from one cell to another when it crosses the boundary between two cells. This may also involve a translation by a multiple of the lattice vectors when using periodic BCs.

Hard-core collision Collide a particle i with a boundary such as a hard wall or another particle j with which it interacts.

Tether collision Bouncing of a pair of tethered particles in a polymer chain when the tether stretches. This collision is processed exactly like usual hard-particle collisions [38, 40].

Time step Move all of the time-driven particles by Δt and process stochastic collisions between them.

The position $\mathbf{r}_i$ and time t_i as well as the impending event prediction of particle i are updated whenever an event involving the particle is processed.

Both the event-driven and the time-driven DSMC algorithms process stochastic binary *trial collisions*. Processing a trial collision consists of randomly and uniformly selecting a pair of DSMC particles i and j that are in the same cell. For hard spheres in the low-density limit, the probability of collision for a particular pair ij is proportional to the relative velocity v_{ij}^{rel}, and therefore the pair ij is accepted with probability $v_{ij}^{\mathrm{rel}}/v_{\mathrm{rel}}^{\mathrm{max}}$. If a pair is accepted for collision than the velocities of i and j are updated in a random fashion while preserving energy and momentum [30]. If a particle i that is in the event queue participates in an actual stochastic collision, then that particle is updated to time t_{TS}, its previous event prediction is invalidated and an immediate update event is scheduled for i. If particle i had a previous scheduled event with a third-party particle k, an immediate update should also be scheduled for particle k.

2. Time Step Events

The hybrid ED/TD algorithm introduces a new kind of *time step event*. This event is scheduled to occur at times $t_{TS} = n\Delta t$, where $n \in \mathcal{Z}$ is an integer. When such an event is processed, all of the DSMC particles not in the event queue are moved [3] to time t_{TS} and are then re-sorted into cells. Note that the ED particles are already correctly sorted into cells. Particles that change from ED to TD cells and vice-versa are removed or inserted into the event queue accordingly.

Next, in each cell $\Gamma_c \Delta t$ trial DSMC collisions are performed, where

$$\Gamma_c = \frac{N_C(N_C-1)\sigma v_{max}}{V_c} \tag{1}$$

is the DSMC collision rate. Here $\sigma = 4\pi R_{DSMC}^2$ in three dimensions and $\sigma = 4R_{DSMC}$ in two dimensions is the collisional cross-section, V_c is the volume of the cell, and v_{max} is an upper bound for the maximal particle velocity [4].

[3] Note that this update may involve moving some particles by less than Δt since the time of the last update for such particles does not have to be a time step event but could be, for example, a cell transfer.

[4] More precisely, $2v_{max}$ is an upper bound on the maximal relative velocity between a pair of particles. In our implementation we maintain the maximal encountered particle velocity v_{max} and update it after every collision and also reset it periodically.

In order to ensure correctness of the AED algorithm, a TD particle must not move by more than a certain distance Δl_{max} when it undertakes a time step. Otherwise, it may overlap with a non-DSMC particle that could not have anticipated this and scheduled a collision accordingly. Specifically, recall that the event-driven cells are marked whenever a neighbor search is performed for a non-DSMC particle. We use

$$\Delta l_{max} = (w_{ED} L_c - D_{DSMC})/2,$$

where the masking width w_{ED} is the minimal number of cells covered by any neighbor search in any direction (typically one or two), L_c is the (minimal) cell length, and D_{DSMC} is the diameter of the DSMC particles. Any DSMC particle whose velocity exceeds $v_{max} = \Delta l_{max}/\Delta t$ is inserted into the event queue at the end of a time step, and similarly, any particles that would have been removed from the event queue are left in the queue if their velocity exceeds the maximum safe velocity. Typically, only a small (albeit non-zero) fraction of the DSMC particles falls into this category and the majority of the particles that are not in ED cells are not in the event queue. In fact, we choose the time step to be as large as possible while still keeping the number of dangerously fast DSMC particles negligible. This typically also ensures that DSMC particles do not jump over cells from one time step to the next.

C. Adaptive Open Boundary Conditions

In three dimensions, a very large number of solvent particles is required to fill the simulation domain. The majority of these particles are far from the polymer chain and they are unlikely to significantly impact or be impacted by the motion of the polymer chain. It therefore seems reasonable to approximate the behavior of the solvent particles sufficiently far away from the region of interest with that of a quasi-equilibrium ensemble. In this ensemble the positions of the particles are as in equilibrium and the velocities follow a local Maxwellian distribution whose mean is the macroscopic local velocity. These particles do not need to be simulated explicitly, rather, we can think of the polymer chain and the surrounding DSMC fluid as being embedded into an infinite reservoir of DSMC particles which enter and leave the simulation domain following the appropriate distributions.

Such *open (Grand Canonical) boundary conditions* (BC) are often used in multi-scale

(coupled) simulations. It is not trivial to implement them when coupling the "reservoir" to an MD simulation, especially at higher densities. An example of an algorithm that achieves such a coupling for soft-particle systems is USHER [8]. It is also non-trivial to account for the velocity distribution of the particles entering the simulation domain [46], as would be needed in a purely event-driven algorithm in which particles are inserted at the surface boundary of the domain. However, the combination of a partially time-driven algorithm and an unstructured (ideal gas) DSMC fluid makes it very easy to implement open BCs by inserting DSMC particles in the cells surrounding the simulation domain only at time-step events, based on very simple distributions.

1. Cell Partitioning

For the purposes of implementing open BCs, we classify the cells as being *interior, boundary, and external cells*. Our implementation uses bits in the cell bitmasks to mark a cell as being event-driven (bit γ_{ED}), boundary (bit γ_B), or external (bit γ_P). The different categories of cells are defined as:

Interior cells are those that are in the vicinity of non-DSMC particles, specifically, cells that are within a window of half-width $w_{int} > w_{ED}$ cells around the centroid of a non-DSMC particle. The interior cells are divided into event-driven and time-driven and are handled as described previously.

Boundary cells surround the interior cells with a layer of cells of thickness $w_B \geq 1$ cells, and they represent cells in which particles may be inserted during time step events. If a boundary or external cell is marked as an event-driven cell due to motion of the non-DSMC particles, then the simulation is aborted with an error. This ensures that particle insertions cannot lead to overlaps with non-DSMC particles.

External cells are non-interior cells that are not explicitly simulated, rather, they provide a boundary condition around the interior and boundary cells. This layer must be at least w_B cells thick, and the cells within a layer of w_B cells around the boundary cells are marked as both external and boundary cells. All of the remaining cells are purely external cells and simply ignored by the simulation.

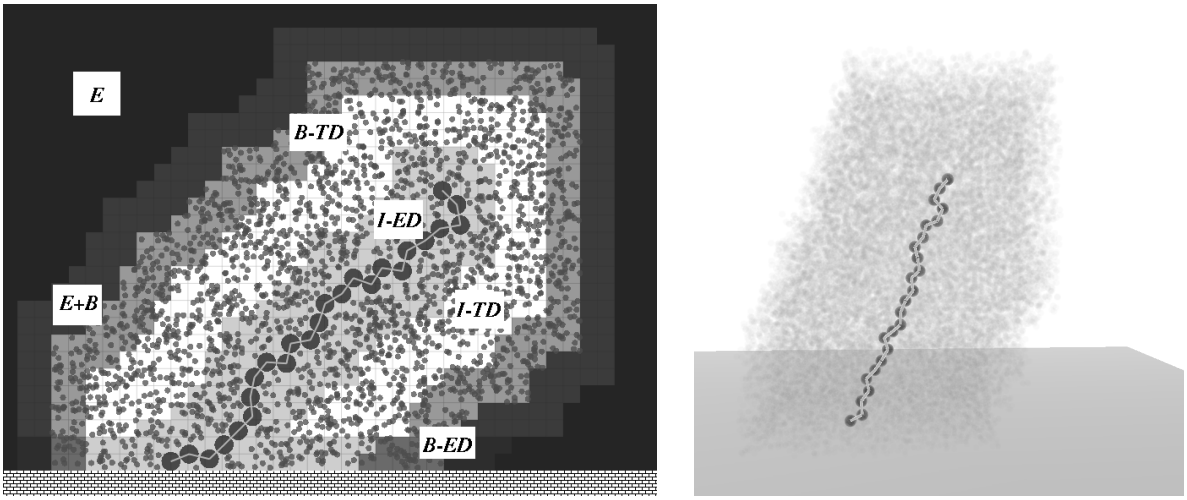

Figure 1: The partitioning of the domain into interior (I) [either event-driven (ED) or time-driven (TD)], boundary (B), and external (E) cells in two (left) and three (right) dimensions for a polymer chain of 25 beads tethered to a hard wall. The cells are shaded in different shades of gray and labeled in the two-dimensional illustration ($w_{ED} = 2$, $w_{int} = 5$, $w_B = 2$). The DSMC particles are also shown.

Note that a cell may be a combination of these three basic categories. In fact, the following types of cells appear in our simulations: (1) event-driven interior (I-ED) cells near non-DSMC particles or hard walls; (2) time-driven interior (I-TD) cells; (3) time-driven boundary (B-TD) cells; (4) event-driven boundary (B-ED) cells, next to hard walls; (5) time-driven boundary external (B+E) cells; (6) external (E) cells not explicitly simulated.

Figure 1 provides an illustration of this division of the cells for the simulation of a tethered polymer in two and three dimensions. Our implementation traverses each of the non-DSMC particles in turn and masks the cells in a window of half-width w cells around the cell containing the non-DSMC particle as interior if $0 \leq w \leq w_{int}$, as boundary if $w_{int} < w \leq w_{int}+2w_B$, and external if $w > w_{int}+w_B$. Here $w_{int} > w_{ED}$ is a chosen extent that covers the region where non-trivial flow occurs. Note that we do not require that the domains of interior or non-external cells form a rectangular domain: The final shapes and even contiguity of such domains depends on the positions of the non-DSMC particles. If this is not appropriate one can always make the simulation regions be unions of disjoint rectangular domains simply by padding with interior cells.

The division of the cells into event-driven, interior, boundary and external cells is rebuilt periodically during the simulation. This rebuilding may only happen at the beginning of time steps, and requires a synchronization of all of the particles to the current simulation time, a complete rebuilding of the cell bitmasks, and finally, a re-initialization of the event processing. Importantly, particles that are in purely external cells are removed from the simulation and those that are in event-driven cells are placed into the event queue scheduled for an immediate update event. During the process of rebuilding the cell bitmasks cells that are masked as purely external cells are also marked as unfilled with the sticky bit γ_s, which is initially not set. This indicates that these cells need to be re-filled with particles later if they enter the simulation domain again due to the motion of the non-DSMC particles. Once the cell bitmasks are rebuilt, a time step event is executed as described next.

2. Time Step Events

When open BCs are used, a time step event consists of the following steps:

1. The time-driven DSMC particles are propagated by time Δt as usual. Those particles that move into purely external cells are removed.
2. The cells that are both external and boundary, or unfilled interior cells, are traversed in order. In each such cell, an appropriate number of trial *reservoir particles* are then inserted and the bit γ_s is reset if the cell was unfilled. For each trial particle:
   (a) The trial particle is propagated by a time step Δt to the current simulation time.
   (b) If the particle moves into a non-external boundary cell, then the trial particle is converted into a real particle.
3. Stochastic collisions are processed in all cells as usual.

In step 2b above a count N_{fast} is kept of the number of trial particles that were not accepted because they moved into a non-boundary cell. If positive, the count N_{fast} is reported at the end of the time-step to aid in choosing w_B sufficiently large so as to ensure that the tails of the velocity distribution are not truncated. In our experience $w_B = 2$ suffices for reasonable choices of Δt.

3. Boundary Conditions

In our current implementation the reservoir particles follow simple local-equilibrium ideal gas distributions. The number of particles to insert in a given cell c is chosen from a Poisson distribution with the appropriate density, the positions are uniformly distributed inside the cell, and the velocities are drawn from a biased (local) Maxwellian distribution. The mean velocity $\mathbf{v}_M$ and temperature T_M for the local Maxwellian are chosen according to the specified boundary conditions, typically uniform linear gradients. For example, if a uniform shear in the xy plane is to be applied, $\mathbf{v}_M = \gamma y_c \hat{x}$, where y_c is the y position of the centroid of the cell and γ is the shear rate. Using such biased local insertions allows one to specify a variety of boundary conditions. For example, a free polymer chain in *unbounded* shear flow can be simulated without resorting to hard-wall boundaries or complicating Lee-Edwards conditions.

It should be noted that in principle we should not use a local Maxwellian velocity distribution for a system that is not in equilibrium. In particular, for small velocity, temperature, and density gradients the Chapman-Enskog distribution is the appropriate one to use in order to avoid artifacts near the open boundaries at length scales comparable to the mean free path λ [47]. We judge these effects to be insignificant in our simulations since our boundary conditions are fixed externally and are thus not affected by the possible small artifacts induced in the DSMC fluid flow, and since λ is small.

In the future, we plan to replace the particle reservoir with a PDE-based (Navier-Stokes) simulation coupled to the DSMC/MD one. Such a flux-preserving coupling has been implemented in the past for coupled DSMC/Euler hydrodynamic simulations [47, 48]. It is however important for the coupling to also correctly couple fluctuations. This requires the use of *fluctuating hydrodynamics* in the coupled domain. Such solvers and associated coupling techniques are only now being developed [26, 27].

D. Further Technical Details

In this section we discuss several technical details of the SEDMD algorithm such as hard-wall boundary conditions and the choice of DSMC parameters.

1. *Slip and Stick Boundary Conditions*

We have already discussed open boundary conditions and their use to specify a variety of "far-field" flow patterns. Additionally, there can also be hard-wall boundaries, i.e., flat impenetrable surfaces. These surfaces can have a velocity of their own and here we discuss how particles reflect from such walls in the frame that moves with the hard wall. Regardless of the details of particle reflections, the total change in linear momentum of all the particles colliding with a hard wall can be used to estimate the friction (drag) force acting on the wall due to the flow. This can give reliable and quick estimates of the viscosity of a DSMC fluid, for example. For hard-wall surfaces, we want to mimic the classical *no-slip* BC. Molecular simulations have found some slip; however, at length-scales significantly larger than the mean free path and/or the typical surface roughness one may assume no-slip boundaries if the hard-wall boundary position is corrected by a slip length L_{slip} [3].

Our simulations of tethered polymers use *thermal walls* [30] kept at $kT = 1$ to implement no-slip hard walls at the boundaries of the simulation cell. Following the collision of a particle with such a wall, the particle velocity is completely randomized and drawn from a half Maxwell-Boltzmann distribution. This automatically ensures a nearly zero mean velocity at the wall boundary and also acts as a thermostat keeping the temperature constant even in the presence of shear heating.

No-slip boundaries can also be implemented using (athermal) *rough walls* which reflect incoming particles with velocity that is the exact opposite of the incoming velocity [49]. Similarly, slip boundary conditions, can be trivially implemented by using *specular walls* that only reverse the normal component of the particle velocity. A mixture of the two can be used to implement partially rough walls, for example, a roughness parameter $0 \leq r_w \leq 1$ can be used as the probability of randomly selecting a rough versus a specular collision.

Similar considerations apply to the boundary conditions at the interface of a hard particle such as a polymer bead. Most particle-based methods developed for the simulation of particle suspensions consider the solvent particles as point particles for simplicity, and only MD or certain boundary discretization schemes [50] resolve the actual solvent-solute interface. Specular BCs are typical of MD simulations and assume perfectly conservative (elastic) collisions. However, if the polymer beads are themselves composed of many atoms, they will act as a partially thermal and rough wall and energy will not be conserved exactly.

In the simulations reported here we have used rough wall BCs for collisions between DSMC and non-DSMC particles. This emulates a non-stick boundary condition at the surface of the polymer beads. Using specular (slip) conditions lowers the friction coefficient [5], but does not appear to qualitatively affect the behavior of tethered polymers.

2. Constant Pressure Gradient Flows

We note briefly on our implementation of constant pressure gradient boundary conditions, as used to simulate flow through open pipes. A constant pressure is typically emulated in particle simulations via a constant acceleration $\mathbf{a}$ for the DSMC particles [51] together with periodic BCs along the flow (acceleration) direction. In time-driven algorithms, one simply increments the velocity of every particle by $\mathbf{a}\Delta t$ and the position by $\mathbf{a}\Delta t^2/2$ at each time-step (before processing DSMC collisions). In SEDMD this is not easily implemented, since the trajectory of the DSMC particles becomes parabolic instead of linear and exact collision prediction between the DSMC and the non-DSMC particles is complicated. We have opted to implement constant pressure BCs by using a periodic delta-function forcing on the DSMC particles. Specifically, the velocities of all DSMC particles are incremented at the beginning of each time step by $\mathbf{a}\Delta t$, and then stochastic collisions are processed. All event-driven DSMC particles are scheduled for an immediate update event because their velocities changed.

3. Choice of DSMC Collision Frequency

The viscosity of the DSMC fluid is determined by the choice of collision frequency Γ_c and cell size L_c. Classical DSMC wisdom [30] is that cell size should be smaller than the mean free path, $L_c \ll \lambda$, but large enough to contain on the order of $N_c \approx 20$ particles (in three dimensions). It is obvious that both of these conditions cannot be satisfied for denser liquids, where λ is only a fraction of the particle size. It is now well-known that it is not necessary to have many particles per cell, so long as in Eq. (1) we use $N_c(N_c-1)$ instead of the traditional (but wrong) N_c^2 [6]. Coupled with the Poisson distribution of N_c this

[5] The Stokes friction force has a coefficient of 4π for slip BCs instead of the well-known 6π for no-slip BCs.
[6] Alternatively, self-collisions can be proposed and rejected.

gives a constant average total collision rate. However, using very small cells leads to very large variability of collision rates from cell to cell and thus spatial localization of momentum transfer during each time step. Namely, very small cells rarely have two or more particles and thus most of the collisions will occur in the few cells that happen to be densely populated.

We have aimed at trying to mimic what would happen in an MD simulation in the DSMC one. In an MD simulation particles collide if their distance is equal to the particle diameter D. Therefore, we have aimed at keeping the cell size at a couple of diameters, $L_c \approx 2D$. At typical hard-sphere liquid densities this leads to $N_c \approx 5 - 10$, which seems appropriate in that it allows enough collision partners for most of the particles but still localized the momentum transfer sufficiently. For very small mean free paths DSMC does not distinguish velocity gradients at length scales smaller than the cell size and, in a long-time average sense, localizes the velocity gradients at cell interfaces [52]. We will assume that the structure of the fluid and flow at length-scales comparable to D (and thus L_c) is unimportant, and verify this by explicit comparisons to MD.

When the cell size is chosen such that $N_c \approx 5 - 10$ and the time step is reasonable, $\Delta t \approx (0.1 - 0.2)L_c/\bar{v}$, Eq. (1) gives collision frequencies that are sufficiently high so that almost all particles suffer at least one collision every time step, and typically more than one collision. The effect of such repeated collisions is to completely thermalize the flow to a local equilibrium (Maxwellian). We have observed that further increasing the collision frequency does not change the effective viscosity and merely wastes computation. We have chosen to use the lowest collision rate that still achieves a viscosity that is as high as using a very high collision rate. We find that this is typically achieved when each particle suffers about half a collision or one collision each timestep [53]. Appendix B describes some multi-particle collision variants that may be more appropriate under different conditions.

4. DSMC without Hydrodynamics

The solvent exerts three primary effects on polymers in flow: (1) stochastic forces due to fluctuations in the fluid (leading to Brownian-like motion), (2) (local) frictional resistance to bead motion, and (3) hydrodynamic interactions between the beads due to perturbations of the flow field by the motion of the beads. Brownian dynamics, the most common method for simulating the behavior of polymers in flow, typically assumes that the drag on the

polymer beads follows Stokes law. Additionally, the first two effects are coordinated via the fluctuation-dissipation theorem. Finally, the third effect is sometimes added via approximations based on the Oseen tensor, neglecting the possibility of large changes to the flow field due to the moving beads.

By turning off local momentum conservation one can eliminate all hydrodynamic interactions, and thus test the importance of the coupling between polymer motion and flow. Yeoman's *et al.* [20, 34] have implemented a no-hydrodynamics variant of the MPCD algorithm by randomly exchanging the velocities between all particles at each time step, thus preserving momentum and energy globally, but not locally. In the presence of a background flow, such as shear, only the components of the velocities relative to the background flow are exchanged. We have implemented a no-hydrodynamics variant of DSMC by neglecting momentum conservation in the usual stochastic binary collisions [7]. Specifically, if particles A and B collide, the post-collisional velocity of A is set to be the same magnitude as the pre-collisional velocity of B but with a random orientation, and vice versa, conserving energy but *not* momentum. If the boundary conditions specify a background flow such as a uniform shear the flow velocity is evaluated at the center of the DSMC cell and the collisions are performed in the frame moving with that velocity. This forces the average velocity profile to be as specified by the boundary conditions, but does not allow for perturbations to that profile due to hydrodynamic effects.

IV. PERFORMANCE IMPROVEMENT

It is, of course, expected that the SEDMD algorithm will give a performance improvement over EDMD. However, to make an impact on real-world problems this performance gain must be an order of magnitude or more improvement. Indeed, we find that SEDMD with adaptive boundary conditions can be up to two orders of magnitude faster than EDMD under certain conditions. Note also that it is well-known that EDMD is already significantly faster than TDMD, although such a comparison is somewhat unfair since the hard-core interaction potentials are very simple by design.

It is not really possible to directly compare SEDMD with EDMD since the two algorithms

[7] In this implementation switching hydrodynamics off becomes an alternative branch localized in the binary collision routine and the algorithm is otherwise unchanged.

utilize different solvents. One is a DSMC solvent in which particles overlap and exchange momentum stochastically, the other is a hard-sphere liquid in which particles do not overlap and collide deterministically. The EDMD fluid is thus more realistic and a direct comparison is somewhat unfair. This is unlike in simulations of equilibrium systems where MC and MD are alternative methods for obtaining equilibrium averages.

The polymer chain is identical in both SEDMD and EDMD simulations, and therefore it is somewhat meaningful to compare the two algorithms based on the total CPU time needed to simulate a unit of simulated time for a single polymer chain in a still solvent. This is the comparison we report here, although the numbers should be interpreted with caution. There are other possibilities, for example, one could compare the CPU time needed to simulate a single relaxation time for the polymer chain. The important point we wish to convey is that SEDMD is one to two orders of magnitude faster than an EDMD simulation, rather than claim exact speedups.

We do not consider or use any parallelization because the EDMD component is very difficult to parallelize scalably. However, because of the inherent simplicity and thus efficiency of the SEDMD algorithm, it is possible to study time scales and system sizes as large or larger than parallel simulations described in the literature so far. The combined time-driven DSMC with event-driven MD algorithm can be parallelized using traditional techniques from TDMD if proper domain partitioning can be constructed, so that each event-driven region is processed by a single processor.

A. Tethered Polymer Chain

As model problem we study a tethered polymer in three dimensions. The solvent density was chosen to be typical of a moderately dense hard-sphere liquid. The performance and optimal choice of parameters depends heavily on the size of the beads relative to the size of the solvent particles for both MD and the hybrid algorithm. Realistically, beads (meant to represent a Kuhn segment) should be larger than the solvent molecules [8]. This of course dramatically increases the computational requirements due to the increase in the number of solvent particles (and also makes neighbor searching more costly). For this reason, most

[8] For example, in Ref. [19] an appropriate bead size for polyethylene is estimated at $1.5nm$, and for DNA (a much stiffer molecule with large persistence length) at $40nm$.

	Standard BCs	Adaptive BCs
Large beads	35	180
Small beads	20	30

Table I: Performance gains of SEDMD relative to EDMD for a typical tethered polymer simulation in three dimensions. The comparison is based on the CPU time needed to simulate a unit of simulation time. As explained in the test, a direct comparison is not really fair or even possible, and therefore these numbers should be interpreted with care.

MD simulations reported in the literature use solvent particles that are equivalent, except for the chain connectivity, to the solute particles.

Our first test problem is for a chain of 25 large beads, each about 10 times larger than the solvent particles in both volume and in mass, in a box of size $2 \times 1.25 \times 1.25$ polymer lengths, for a total of about $N = 2.3 \times 10^5$ particles. For the SEDMD simulations, we did not use bounding sphere complexes (BSCs) [39], and therefore the neighbor search had to include next-nearest neighbor cells as well (i.e., $w_{ED} = 2$). For the corresponding MD simulations, BSCs were used. Under these conditions, SEDMD outperformed EDMD by a factor of 35. If adaptive open BCs were used with $w_{int} = 5$, giving about $N = 3.2 \times 10^4$ particles, the speedup was 180. While this may seem an unfair comparison, it is important to point out that it is not clear how to implement an adaptive simulation domain in pure EDMD.

The second test problem was for a chain of 30 beads which were identical to the solvent particles, except for the added chain tethers. The number of particles in the simulation cell was thus much smaller, $N = 4.8 \times 10^4$, and $w_{ED} = 1$. Adaptive BCs with $w_{int} = 5$ reduce the simulation domain to $N = 2.2 \times 10^4$ particles. For these parameters SEDMD with adaptive BCs was about 30 times faster than EDMD. Table I summarizes the large performance gains of SEDMD relative to traditional EDMD.

One of the fundamental problems with multi-scale modeling is that typically the majority of the simulation time is spent in the finest model since it is difficult to match the time scales of the coupled components [24]. For example, MD simulations are so expensive that coupling them to almost any meso- or macro-scopic solver leads to simulation times limited by that of MD simulations, albeit of a much smaller system. By virtue of the fast microscopic algorithm (EDMD instead of TDMD) and the efficient coupling, our method spends compa-

rable amounts of computation on the solute and immediately surrounding solvent, and on the solvent particles. For the DSMC run with adaptive open BCs and large beads, about 50% of the time was spent in manipulation of near neighbor lists. Most of the remaining time was spent inside the routine that takes a DSMC timestep, and actual processing of DSMC collisions (both trial and real) occupied about 20% of the total computation time. For small beads, the majority of the time, 80%, was spent in the DSMC time-step routine, and processing of DSMC binary collisions occupied about 35% of the total computation time.

V. TETHERED POLYMER IN SHEAR FLOW

In this section results are presented for a tethered polymer chain in uniform shear in three dimensions. The linear chain is in a good solvent and is attached at one end to a hard wall, as represented by the plane $y = 0$. A linear velocity profile $v = \gamma y\hat{x}$ along the x axis is imposed sufficiently far from the chain. This problem was first studied experimentally by Doyle *et al.* [7] and since then numerous computational studies have investigated various aspects of the problem [8–11, 13]. We will focus on the dynamics of the chain at low to medium flow rates (i.e., small Weissenberg numbers) because we wanted to verify that our polymer and solvent model can correctly reproduce non-trivial dynamics.

A. Background

The properties of a linear polymer in shear flow can be related to the dimensionless Weissenberg number $\mathrm{Wi} = \gamma\tau_0$, where $\tau_0 = \tau(\gamma = 0)$ is the relaxation time of the polymer chain when there is no shear. When $\mathrm{Wi} < 1$ the flow barely affects the polymer, contrary to when $\mathrm{Wi} > 1$. Different models have given similar properties for the same Weissenberg number.

The original experimental study of tethered polymers [7] observed what was termed "cyclic dynamics" of the chains. Specifically, the following cycle was proposed. When the polymer moves too far from the wall, presumably by an unusual fluctuation, it experiences a stronger flow and is stretched. A torque develops that then pushes the chain closer to the wall, where it can contract again due to the weaker flow near the wall. The cycle then repeats. Experiments [7] did not identify clear periodicity of this motion. Subsequent computational

studies have looked for such a characteristic period for this cycling motion.

The MD study in Ref. [9] examined the cross-correlation function $C_{X\phi}(t)$, where X measures the extension of the polymer along the flow, and ϕ measures the angle of the chain with respect to the hard wall. No exact definitions of X or ϕ were given even though there are several possibilities. One can use the difference between the maximal and the minimal bead positions as a measure of the extension along a given axes. Optionally, one can simply use the maximal position, or one can use the position of the last bead. Similarly, the angle of the polymer can be based on a linear fit to the shape of the chain, on the position of the center of mass, the asymmetry of the gyration tensor [12], or the position of the last bead. We have examined various choices and have found little qualitative difference between the different choices. We have found the position of the end bead $\mathbf{r}_{N_b} = (x, y, z)$ to be the best option and will also measure the angle $\phi = \tan^{-1}(y/x)$.

The authors of Ref. [9] found that $C_{x\phi}(t)$ develops a peak at positive time t^* for sufficiently large Wi numbers (Wi > 10). This was interpreted as supporting the existence of a critical Weissenberg number Wi where the flow effect on the polymer dynamics changes qualitatively. It was also found that t^* decreases with increasing Wi and the height of the peak increases. It is important to note that t^* was found to be comparable to the relaxation time of the polymer τ_0. Additionally, the internal relaxation time τ was found to decrease with increasing Wi, in agreement with theoretical predictions.

A subsequent study which used a hybrid MD/CFD model, and also a (free-draining) Brownian dynamics model, claimed to observe periodic oscillations in the cross-correlation function between the extensions along the flow and along the shear direction, $C_{xy}(t)$ [10, 13]. However, the period of oscillation was found to be an order of magnitude larger than the internal relaxation time, as revealed by a small peak in the power spectral density $PSD_{xy}(f)$ of $C_{xy}(t)$. A similar claim was made in Ref. [12] based on $PSD_{\phi\phi}$ of the polymer angle autocorrelation function [9] $C_{\phi\phi}(t)$ for both a free polymer in unbounded shear flow and a tethered polymer in shear flow. No results for the short-time cross-correlation functions were reported in either of these studies making it difficult to reconcile the results obtained from PSDs with those in Ref. [9].

[9] The PSD is equivalent to the Fourier spectrum power of the angle trace $\phi(t)$ based on the convolution theorem.

Most experimental and computational studies of the dynamics of polymers in shear flow have been for free chains in unbounded flow [4]. In that problem, for $\mathrm{Wi} > 1$, it is possible to identify a well-defined "tumbling" event as the polymer rotates. The frequency of such tumbling times can be measured by visual inspection and have been compared to the computed location of the peak in the PSDs [12, 54]. The good match has thus been taken as an indicator that PSDs peaks can be used to determine characteristic tumbling times and the same methodology has been applied to a tethered polymer as well. However, for the case of a tethered chain it is not easy to identify a periodic event such as a specific rare fluctuation. Therefore, it is not surprising that we do not confirm the existence of a characteristic time that is an order of magnitude larger than the internal relaxation time. One must here distinguish between "cyclic" (repetitive) events and periodic events. A Poisson time process of rate Γ has a well-defined time scale Γ^{-1}, however, the occurrence of such events is not periodic; the delay between successive events is exponentially-distributed. In Ref. [54] such an exponential distribution is proposed even for the delay between successive tumbling events for a free chain in unbounded flow. The PSD of such a process is expected to be that of white noise (i.e., flat) for frequencies small compared to Γ, and typically a power-law decay for larger frequencies (gray noise). The occurrence and shape of any local maxima (peaks) or frequencies comparable to Γ depends on the exact nature of the correlations at that time scale.

B. Model Parameters

As explained in IV, we have made several runs for different polymer lengths and also bead sizes. One set of runs used either $N_b = 25$ or 50 large beads each about 10 times larger than a solvent particle, using DSMC with or without hydrodynamics (see Section III D 4) for the solvent. Another set of runs used either $N_b = 30$ or 60 small beads each identical to a solvent particle, using DSMC or pure MD for the solvent. The beads were rough in the sense that no-slip conditions were applied for the solvent-solute interface (see Section III D 1).

All of the runs used open boundary conditions (see Section III C), and the typical half-width of the interior region was $w_{int} = 5$ or $w_{int} = 7$ cells around the polymer chain. The difference in the results, such as relaxation times, between these runs and runs using

$w_{int} = 10$ or runs using periodic BCs were negligible for the chain sizes we studied [10]. The solvent was a hard-sphere MD or DSMC fluid with volume fraction $\phi \approx 0.25 - 0.30$, which corresponds to a moderately dense liquid (the melting point is $\phi_m \approx 0.49$). The $N_b = 30$ runs were run for $T \approx 6000\tau_0$ with $w_{int} = 7$, and such a run takes about 6 days on a single 2.4GHz Dual-Core AMD Opteron processor. Even for such long runs the statistical errors due to the strong fluctuations in the polymer conformations are large, especially for correlation functions at long time lags $t > \tau$.

C. Relaxation Times

The *relaxation time* of the polymer τ is well-defined only for linear models. It is often measured by fitting an exponential to the autocorrelation function of the end-to-end vector $\mathbf{r}_{end}(t) = \mathbf{r}_{N_b} - \mathbf{r}_1$, where $\mathbf{r}_i$ denotes the position of the i-th bead [6]. We will separately consider the different components of the end-to-end vector $\mathbf{r}_{end} = (x, y, z)$ and fit an exponential to the C_{xx}, C_{yy} and C_{zz} auto-correlations functions to obtain the relaxation times τ_x, τ_y and τ_z as a function of Wi. The initial relaxation of the various auto-correlation functions $C(t)$ is faster than exponential, and the statistical error at longer times is large even for long runs. We therefore fit the exponentials to the portion of the curves at small times, when $0.2 \leq C(t) \leq 0.8$. The fits are not perfect and there are large statistical errors depending on the length of the run and the number of samples used to average $C(t)$, and the relaxation times and Weissenberg numbers we quote should be taken as approximate.

We find that τ_z is always the largest, especially for large Wi (for Wi $= 0$, $\tau_z = \tau_x$ by symmetry), and τ_y is always smaller by at least a factor of two [11], even for Wi $= 0$, as illustrated in the inset in Fig. 2. We take $\tau_0 = \tau_x(\text{Wi} = 0) = \tau_z(\text{Wi} = 0)$ as the definition of the polymer relaxation time. Figure 2 illustrates the dependence of $\tau_x(\text{Wi})/\tau_x(\text{Wi} = 0)$ on Wi, and similarly for the y and z directions. Quantitatively similar (but not identical) results are observed independently of the details of the polymer model and even the existence

[10] It is expected that using a small w_{int} would truncate the (long-ranged) hydrodynamic interactions and thus increase the relaxation time. We observe such effects for the $N_b = 50$ chains, however, the effect is too small compared to the statistical errors to be accurately quantified.

[11] This is because of the constraint that the polymer chain must be above the plane $y = 0$ at all times, which reduces the available configuration space.

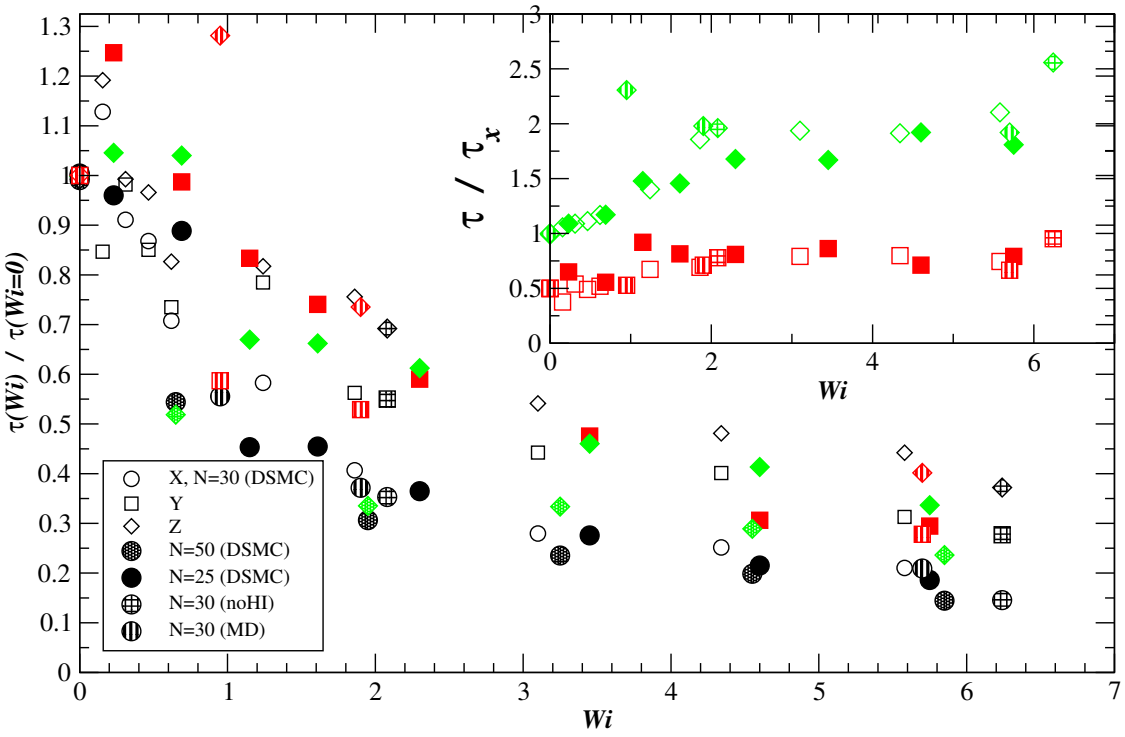

Figure 2: Dependence of the relaxation times of the different components of the end-to-end displacement vector on the Weissenberg number. The relaxation times have been renormalized to equal unity for Wi = 0 for direct comparison. For each Wi, τ_x is shown with circles, τ_y with squares, and τ_z with diamonds. Different textures of the symbols are used for the different models, as indicated in the legend. The inset shows τ_y/τ_x and τ_z/τ_x for the different runs.

of hydrodynamic interactions.

The relaxation times we observe for Wi = 0 are consistent with what is predicted from theoretical considerations, $\tau \approx 0.9\eta b^3 N_b^{1.8}/kT$, where η is the viscosity and b is the effective bead radius. Direct measurements of the viscosity of the DSMC liquid show that it has viscosity rather close to that of the corresponding MD liquid for the specific parameters we use. Using the Enskog viscosity of the MD liquid and the tether length as b, we calculated $\tau \approx 19$ for the case of $N_b = 25$ with large beads, to be compared to the numerical results from DSMC $\tau = 25 \pm 5$. The MD runs for the case of large beads are not long enough to determine the relaxation time accurately. We expect that the difference between MD and DSMC will become more pronounced for smaller beads, and indeed, for $N_b = 30$ we obtain $\tau_{MD} \approx 3\tau_{DSMC}$.

Turning hydrodynamics off in DSMC extends the relaxation times (and also the collapse

times for an initially stretched polymer) by a factor of $3-5$, as already observed using MPCD [20] and as predicted by Zimm theory. It is difficult to directly compare DSMC with and without hydrodynamics since switching hydrodynamics off, in our model, affects the friction force between the beads and the solvent. This is unlike the models were the friction force is an added phenomenological term that has an adjustable coefficient.

D. Cyclic Dynamics

We now turn our attention to cross-correlations between polymer extensions in the x and y directions. We have found that the cross-correlations lags are most visible in the x and y positions of the last bead, $C_{xy}(t)$. Our results for $C_{xy}(t)$ are shown in Fig. 3, along with $C_{x\phi}(t)$ as an inset. The results for $C_{x\phi}(t)$ compare well with those in Ref. [9], although we see the secondary peak developing at somewhat lower Wi. We do not see any evidence for the existence of a critical Wi: There are peaks at both positive and negative time in $C_{xy}(t)$ for all Wi. Some cross-correlations, such as $C_{x\phi}(t)$, have a large positive or negative cusp at the origin at $\mathrm{Wi}=0$ and it is this cusp that masks the peaks at non-zero lags for small Wi.

In Fig. 4 we compare $C_{xy}(t)$ at $\mathrm{Wi}\approx 2$ for several different models [12] and see a good match, even for the DSMC runs ignoring hydrodynamics (momentum conservation). This indicates that the dynamics of the chains is primarily driven by the competition between the internal stochastic motion (entropy) and the external forcing due to the shear, and not hydrodynamic interactions between the beads or the effect of the motion of the chain on the flow.

We do not discuss the origin and locations of the peaks in the cross-correlation functions in detail in this work. These peaks are indicative of the existence of a correlated motion in the xy plane, but do not uniquely identify that motion. An important question to address is the existence of a time scale other than the internal relaxation time $\tau(\mathrm{Wi})$. In Fig. 5 we show a renormalized cross-correlation function

$$\tilde{C}_{xy}=\frac{1}{\mathrm{Wi}}C_{xy}\left[\frac{t}{\tau(\mathrm{Wi})}\right]$$

in an unsuccessful attempt to collapse the data for different Wi. While the match is not

[12] The Weissenberg numbers were calculated after the runs were completed and therefore the different runs are not at the exact same Wi number.

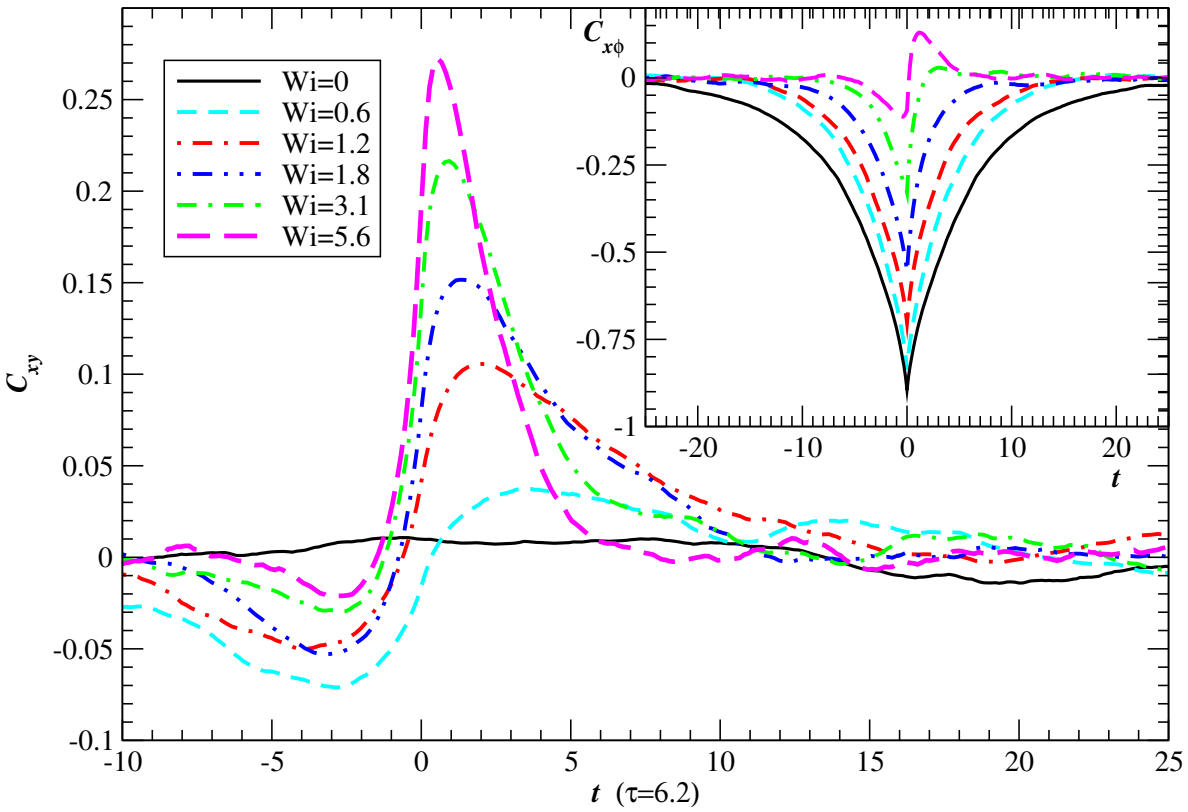

Figure 3: Cross correlation function $C_{xy}(t)$ for chains of $N_b = 30$ small beads in a DSMC solvent at different shear rates. The inset shows the corresponding $C_{x\phi}(t)$ for comparison with the soft-particle MD results in Ref. [9]. Peaks are visible in $C_{xy}(t)$ at all Wi > 0, but are obscured in $C_{x\phi}(t)$ due to the large negative dip at the origin for Wi $= 0$. There are large statistical errors at small Wi making it difficult to identify the peaks.

perfect the picture does not point to the existence of a time scale shorter than τ(Wi). We also do not see any convincing evidence for coherent and reproducible correlations on time scales significantly larger than τ, even in various power spectral densities. Our results do not rule out the possibility of a repetitive motion of the chain with widely varying cyclic times (e.g., exponential tail) but we have not observed any direct evidence for such cycling either. We will report more detailed results on the dynamics of tethered polymer chains along with comparisons with Brownian dynamics and Lattice-Boltzmann in future work.

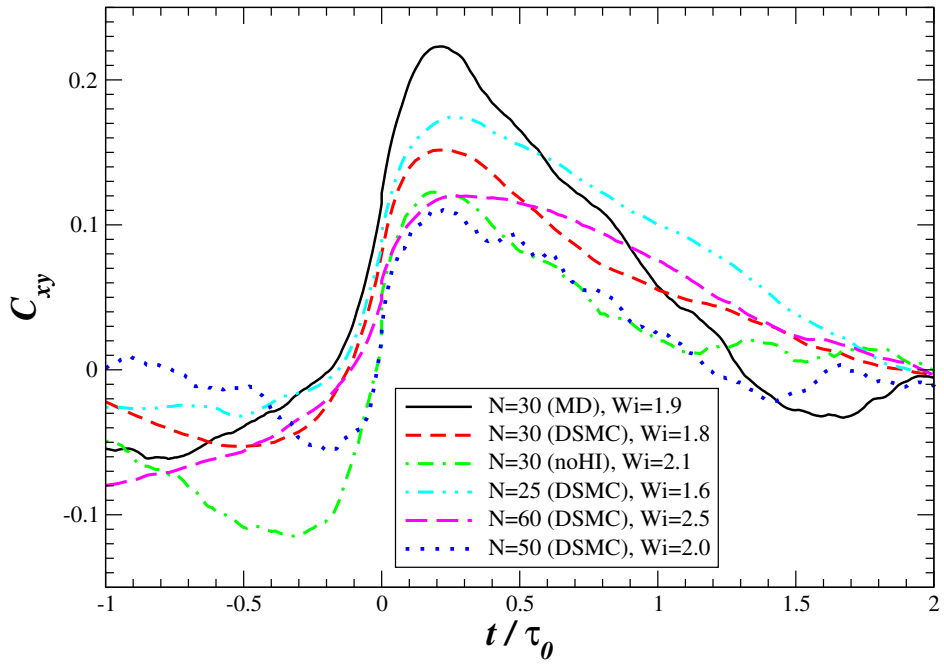

Figure 4: Comparison of $C_{xy}(t)$ for Weissenberg number of about 2 for several different models, after the time axes has been normalized.

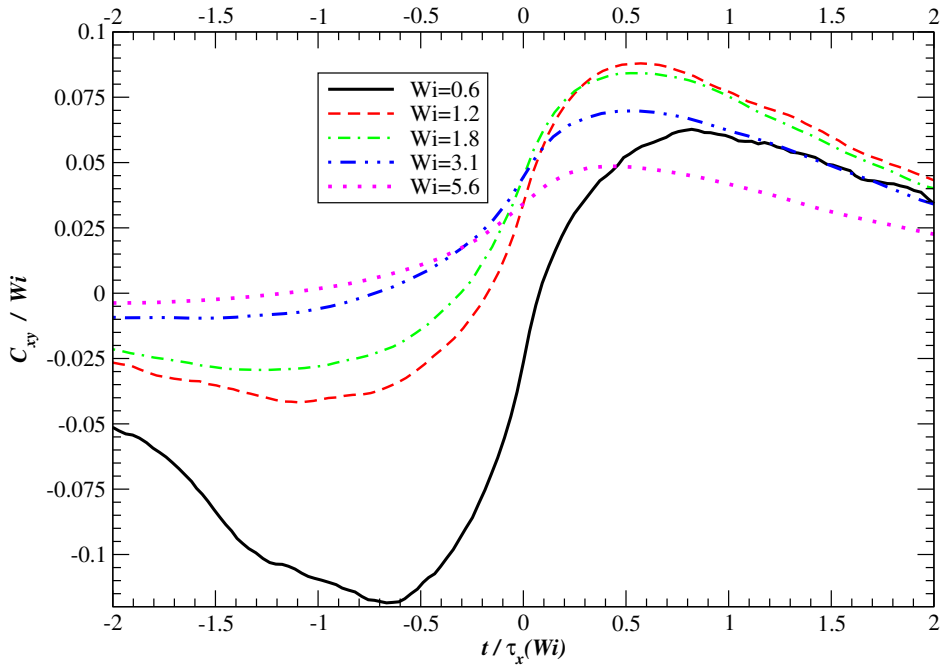

Figure 5: Cross correlation function $\tilde{C}_{xy}$ for $N = 30$ DSMC runs as in Fig. 3 but with time renormalized by τ(Wi) and the correlation magnitude scaled by Wi.

VI. CONCLUSIONS

We presented a stochastic event-driven molecular dynamics (SEDMD) algorithm that combines hard-sphere event-driven molecular dynamics (EDMD) with direct simulation Monte Carlo (DSMC), aimed at simulating flow in suspensions at the microscale. The overall algorithm is still event-driven, however, the DSMC portion of the algorithm can be made time-driven for increased efficiency. The fundamental idea is to replace the deterministic (MD-like) interactions between particles of certain species with a stochastic (MC-like) collision process, thus preserving the phase space dynamics and conservation laws but ignoring the liquid structure. The SEDMD methodology correctly reproduces hydrodynamic behavior at the macroscale but also correctly represents fluctuations at the microscale. A similar algorithm has been proposed using time-driven (soft-particle) MD and a multiparticle collision variant of DSMC [23].

As an application of such a methodology we have considered the simulation of polymer chains in a flowing solution, and in particular, a polymer tethered to a hard wall and subject to shear flow. We have implemented open boundary conditions that adaptively adjust the simulation domain to only focus on the region close to the polymer chain(s). The algorithm is found to be efficient even though it is not parallelized, and it is found to reproduce results obtained via molecular dynamics and other algorithms in the literature, after adjusting for the correction to transport coefficients and compressibility of the DSMC fluid relative to the MD fluid.

We studied the dynamics of a tethered polymer subject to pure shear and found consistent results between EDMD and SEDMD and also previous TDMD studies. We find that neither the size of the polymer beads relative to the solvent particles, nor the correct representation of the hydrodynamic interactions in the fluid, qualitatively alter the results. This suggests that fluctuations dominate the dynamic behavior of tethered polymers, consistent with previous studies. Our results do not find periodic motion of the polymer and show that the cross-correlation between the polymer extensions along the flow and shear directions shows a double-peak structure with characteristic time that is comparable to the relaxation time of the polymer. This is in contrast to other works that claim the existence of a new timescale associated with the cyclic motion of the polymer. We will investigate these issues further and compare with Brownian dynamics and Lattice-Boltzmann simulations in future work.

We expect that this and related algorithms will find many applications in micro- and nano-fluidics. In particular, the use of DSMC instead of expensive MD is suitable for problems where the detailed structure and chemical specificity of the solvent do not matter, and more general hydrodynamic forces and internal fluctuations dominate. Using a continuum approach such as Navier-Stokes (NS) equations for the solvent is questionable at very small length scales. Furthermore, the handling of singularities and fluctuations is not natural in such PDE methods and various approximations need to be evaluated using particle-based methods. Since the meshes required by continuum solvers for microflows are very fine, it is expected that the efficiency of particle methods will be comparable to PDE solvers. Nevertheless, algorithms based on fluctuating hydrodynamics descriptions will be more efficient when fluctuations matter. Comparisons and coupling of DSMC to fluctuating NS solvers is the subject of current investigations [27].

VII. ACKNOWLEDGMENTS

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (UCRL-JRNL-233235).

Appendix A: AED VARIANTS OF DSMC

In this Appendix we discuss a fully asynchronous event-driven (AED) implementation of DSMC. The advantage of asynchronous algorithms is that they do not introduce any artificial time scales (such as a time step) into the problem [44]. We have validated that the AED algorithm produces the same results as the time-driven one by comparing against published DSMC results for plane Poiseuille flow of a rare gas [51]. We have also implemented traditional time-driven (TD) DSMC and find identical results when the time step is sufficiently small. We find that the event-driven algorithm is almost an order of magnitude slower than the time driven one at higher densities, and only becomes competitive at very low densities, which is the traditional domain of interest for DSMC. The overhead of the AED algorithm comes from the need to re-predict the next event and update the event queue whenever a particle suffers a DSMC collision. This cost is in addition to the equivalent cost in the

time-driven algorithm, namely, moving the particles forward in time and colliding them.

The AED algorithm introduces a new type of event, a *stochastic (trial) collision* between two DSMC particles that are in the same cell (see Section III B). These trial collisions occur in a given cell c as a Poisson process with a rate given by Eq. (1). There are several approaches to scheduling and processing DSMC collisions directly borrowed from algorithms for performing Kinetic Monte Carlo simulations, which are *synchronous* event-driven algorithms [55]. The simplest, and in our experience, most efficient, approach to AED DSMC is to use *cell rejection* to select a host cell for the stochastic collisions. The rate of DSMC collisions is chosen according to the cell with maximal occupancy N_c^{max}, $\Gamma = N_{cells}\Gamma_c^{max}$. The randomly chosen cell c of occupancy N_c is accepted with probability $N_c(N_c-1)/\left[N_c^{max}(N_c^{max}-1)\right]$ and a random pair of particles i and j are chosen from $\mathcal{L}_c$. Since the DSMC fluid is perfectly compressible, the maximal cell occupancy can be quite high for very large systems, and this leads to decreasing cell acceptance probability as the size of the system increases.

One can avoid cell rejections altogether. The first option is to associate stochastic collisions with cells and schedule one such *collision-in-cell* event per cell. The event time is easily predicted at any point in time t to occur at time $t - \Gamma_c^{-1}\ln r$, where r is a uniform random deviate in $(0,1)$. These event times are put in an event queue, which may be the same as the EDMD event queue, or it may be separate queue then the two queues may be merged only at the top. The collision-in-cell event times need to updated whenever a cell occupancy N_c changes, that is, whenever a cell transfer is processed. This makes this algorithm inefficient. Another alternative is to recognize that the sum of a set of independent Poisson processes is a Poisson process with a rate that is the sum of the individual rates, $\Gamma = \sum_c \Gamma_c$. That is, DSMC collisions occur in the system as a Poisson process with rate Γ. When processing such an event one has to first choose the cell with probability Γ_c/Γ, which requires some additional data structures to implement efficiently [55]. For example, the cells could be grouped in lists based on their occupancy and then an occupancy chosen first with the appropriate weight, followed by selection of a cell with that particular occupancy.

Finally, it is also possible to use a mixture of the asynchronous and time-driven variants of DSMC. The asynchronous algorithm can be used for DSMC particles in event-driven cells, and the time-driven one elsewhere. This may be useful in situations where the time-scale of the event-driven component is significantly smaller than the time step Δt and thus time

stepping would lead to discretization artifacts.

In the AED variant of DSMC constant pressure BCs (see Section III D 2) can be implemented by adding a new type of *acceleration event*. When such an event is processed, all of the particles are brought to the same point in time (synchronized), the velocities of each DSMC particle i is incremented by $\mathbf{a}\Delta t_i$, where Δt_i is the elapsed time since the last acceleration event. Following an acceleration event, the event queue is reset because all of the event predictions are invalidated by the change in particle velocities. The acceleration events occur as a Poisson process with a suitably chosen rate, for example, ensuring that the average or maximal change in velocity is a fraction of the average particle velocity. Note that the choice of this acceleration rate introduces an artificial time constant in the algorithm similar to the time step Δt in time-driven DSMC.

Appendix B: MULTI-PARTICLE COLLISIONS IN DSMC

Under dense liquid conditions, DSMC binary collisions are so numerous (see Section III D 3) that the velocities of the particles are effectively thermalized to the local Maxwell distribution. We have implemented a variant DSMC algorithm in which at every time step the velocities of all of the particles are redrawn from a local Maxwellian, preserving the total linear momentum and energy in each cell [56]. We found that this variant of DSMC is less efficient than and behaves similarly to the usual binary-collision DSMC. Reference [57] describes a more general algorithm (TRMC) that combines binary collisions for a subset of the particles with drawing from a local Maxwellian for the remainder of the particles, and under dense liquid conditions this typically degenerates to complete randomization of all of the velocities at every time step. Until a theoretical framework is established for the behavior of DSMC-like algorithms at high densities the classical DSMC algorithm seems to be the best alternative in terms of simplicity, efficiency, and theoretical foundation.

We mention that, strictly speaking, we should use as V_c in Eq. (1) not the volume of the cell, but the unoccupied cell volume, i.e., the portion of the cell not covered by non-DSMC particles [13]. It is however difficult to dynamically maintain an accurate estimate of

[13] Our implementation makes the additional approximation that non-DSMC particles are also counted in N_c in Eq. (1), instead of keeping a separate count of just the DSMC particles inside each cell. If the polymer beads are larger than a cell than this approximation does not matter since no cell can contain

the cell coverage, and the complication does not appear to be worth the implementation complexity. In particular, an approximation is already made in neglecting the structure of the solvation layer around a polymer bead. Furthermore, the majority of the cells that are partially covered by a polymer bead will be entirely or almost entirely covered so that they would at most contain a single DSMC particle, in which case the probability of a DSMC collision would be very low anyway. Finally, as explained in Section III D 3, the exact collision frequency does not really matter. In the context of multiparticle collision dynamics, Ref. [49] proposes the use of virtual particles filling the partially-filled cells as a way to achieve more accurate stick boundary conditions.

---

[1] W. E. Alley, P. Covello, and B. J. Alder. Complex flows by nanohydrodynamics. *Mol. Phys.*, 102(19):0026–8976, 2004.

[2] M. Ripoll, K. Mussawisade, R. G. Winkler, and G. Gompper. Low-Reynolds-number hydrodynamics of complex fluids by multi-particle-collision dynamics. *Europhys. Lett.*, 68(1):106, 2004.

[3] T. M. Squires and S. R. Quake. Microfluidics: Fluid physics at the nanoliter scale. *Rev. Mod. Phys.*, 77(3):977, 2005.

[4] E. S. G. Shaqfeh. The dynamics of single-molecule DNA in flow. *Journal of Non-Newtonian Fluid Mechanics*, 130:1–28, 2005.

[5] G. De Fabritiis, M. Serrano, R. Delgado-Buscalioni, and P. V. Coveney. Fluctuating hydrodynamic modeling of fluids at the nanoscale. *Phys. Rev. E*, 75(2):026307, 2007.

[6] G. W. Slater, Y. Gratton, M. Kenward, L. McCormick, and F. Tessier. Deformation, Stretching, and Relaxation of Single-Polymer Chains: Fundamentals and Examples. *Soft Materials*, 2:155–182, 2004.

[7] P. S. Doyle, B. Ladoux, and J.-L. Viovy. Dynamics of a Tethered Polymer in Shear Flow. *Phys. Rev. Lett.*, 84(20):4769–4772, 2000.

[8] S. Barsky, R. Delgado-Buscalioni, and P. V. Coveney. Comparison of molecular dynamics with hybrid continuum–molecular dynamics for a single tethered polymer in a solvent. *J. Chem.*

the centroid of both a DSMC and non-DSMC particle.

Phys., 121(5):2403–2411, 2004.

[9] Y. Gratton and G. W. Slater. Molecular dynamics study of tethered polymers in shear flow. *Eur. Phys. J. E*, 17:455–465, 2005.

[10] R. Delgado-Buscalioni. Cyclic Motion of a Grafted Polymer under Shear Flow. *Phys. Rev. Lett.*, 96(8):088303, 2006.

[11] G. M Wang and W. C Sandberg. Non-equilibrium all-atom molecular dynamics simulations of free and tethered DNA molecules in nanochannel shear flows. *Nanotechnology*, 18(13):135702, 2007.

[12] C. M. Schroeder, R. E. Teixeira, E. S. G. Shaqfeh, and S. Chu. Characteristic Periodic Motion of Polymers in Shear Flow. *Phys. Rev. Lett.*, 95(1):018301, 2005.

[13] R. Delgado-Buscalioni. Dynamics of a Single Tethered Polymer under Shear Flow. *AIP Conference Proceedings*, 913(1):114–120, 2007.

[14] R. M. Jendrejack, J. J. de Pablo, and M. D. Graham. Stochastic simulations of DNA in flow: Dynamics and the effects of hydrodynamic interactions. *J. Chem. Phys.*, 116(17):7752–7759, 2002.

[15] J. P. Hernandez-Ortiz, J. J. de Pablo, and M. D. Graham. Fast Computation of Many-Particle Hydrodynamic and Electrostatic Interactions in a Confined Geometry. *Phys. Rev. Lett.*, 98(14):140602, 2007.

[16] O. B. Usta, A. J. C. Ladd, and J. E. Butler. Lattice-Boltzmann simulations of the dynamics of polymer solutions in periodic and confined geometries. *J. Chem. Phys.*, 122(9):094902, 2005.

[17] N. Sharma and N. A. Patankar. Direct numerical simulation of the Brownian motion of particles by using fluctuating hydrodynamic equations. *J. Comput. Phys.*, 201:466–486, 2004.

[18] D. Trebotich, G. H. Miller, P. Colella, D. T. Graves, D. F. Martin, and P. O. Schwartz. A Tightly Coupled Particle-Fluid Model for DNA-Laden Flows in Complex Microscale Geometries. *Comp. Fluid Solid Mech.*, pages 1018–1022, 2005.

[19] G. Giupponi, G. De Fabritiis, and P. V. Coveney. Hybrid method coupling fluctuating hydrodynamics and molecular dynamics for the simulation of macromolecules. *J. Chem. Phys.*, 126(15):154903, 2007.

[20] N. Kikuchi, J. F. Ryder, C. M. Pooley, and J. M. Yeomans. Kinetics of the polymer collapse transition: The role of hydrodynamics. *Phys. Rev. E*, 71(6):061804, 2005.

[21] D. Trebotich, G. H. Miller, and M. D. Bybee. A Hard Constraint Algorithm to Model Particle

Interactions in DNA-laden Flows. *Nanoscale and Microscale Thermophysical Engineering*, 11(1):121–128, 2007.

[22] K. Mussawisade, M. Ripoll, R. G. Winkler, and G. Gompper. Dynamics of polymers in a particle-based mesoscopic solvent. *J. Chem. Phys.*, 123(14):144905, 2005.

[23] S. H. Lee and R. Kapral. Mesoscopic description of solvent effects on polymer dynamics. *J. Chem. Phys.*, 124(21):214901, 2006.

[24] D. Trebotich, G. H. Miller, and M. D. Bybee. A Penalty Method to Model Particle Interactions in DNA-laden Flows. To appear in J. Nanosci. Nanotech., 2007.

[25] C. Aust, M. Kroger, and S. Hess. Structure and dynamics of dilute polymer solutions under shear flow via nonequilibrium molecular dynamics. *Macromolecules*, 32(17):5660–5672, 1999.

[26] J. B. Bell, A. Garcia, and S. A. Williams. Numerical Methods for the Stochastic Landau-Lifshitz Navier-Stokes Equations. *Phys. Rev. E*, 76:016708, 2007.

[27] S. A. Williams, J. B. Bell, and A. L. Garcia. Algorithm Refinement for Fluctuating Hydrodynamics. Submitted, 2007.

[28] K. Kadau, C. Rosenblatt, J. L. Barber, T. C. Germann, Z. Huang, P. Carles, and B. J. Alder. The importance of fluctuations in fluid mixing. *PNAS*, 104(19):7741–7745, 2007.

[29] F. Xijunand N. Phan-Thien, S. Chen, X. Wu, and T. Y. Ng. Simulating flow of DNA suspension using dissipative particle dynamics. *Physics of Fluids*, 18(6):063102, 2006.

[30] F. J. Alexander and A. L. Garcia. The Direct Simulation Monte Carlo Method. *Computers in Physics*, 11(6):588–593, 1997.

[31] F. Baras, M. Malek Mansour, and A. L. Garcia. Microscopic simulation of dilute gases with adjustable transport coefficients. *Phys. Rev. E*, 49(4):3512–3515, 1994.

[32] F. J. Alexander, A. L. Garcia, and B. J. Alder. A Consistent Boltzmann Algorithm. *Phys. Rev. Lett.*, 74(26):5212–5215, 1995.

[33] A. L. Garcia and W. Wagner. The limiting kinetic equation of the Consistent Boltzmann Algorithm for dense gases. *J. Stat. Phys.*, 101:1065–86, 2000.

[34] J. F. Ryder and J. M. Yeomans. Shear thinning in dilute polymer solutions. *The Journal of Chemical Physics*, 125(19):194906, 2006.

[35] Jose Maria Montanero and Andres Santos. Simulation of the Enskog equation [a-grave] la Bird. *Phys. Fluids*, 9(7):2057–2060, 1997.

[36] A. Frezzotti. A particle scheme for the numerical solution of the Enskog equation. *Phys.*

Fluids, 9(5):1329–1335, 1997.

[37] T. Ihle, E. Tüzel, and D. M. Kroll. Consistent particle-based algorithm with a non-ideal equation of state. *Europhys. Lett.*, 73:664–670, 2006.

[38] B. J. Alder and T. E. Wainwright. Studies in molecular dynamics. I. General method. *J. Chem. Phys.*, 31:459, 1959.

[39] A. Donev, S. Torquato, and F. H. Stillinger. Neighbor List Collision-Driven Molecular Dynamics Simulation for Nonspherical Particles: I. Algorithmic Details II. Applications to Ellipses and Ellipsoids. *J. Comp. Phys.*, 202(2):737–764, 765–793, 2005.

[40] S. W. Smith, C. K. Hall, and B. D. Freeman. Molecular Dynamics for Polymeric Fluids Using Discontinuous Potentials. *J. Comp. Phys.*, 134(1):16–30, 1997.

[41] S. B. Opps, J. M. Polson, and N. A. Risk. Discontinuous molecular dynamics simulation study of polymer collapse. *J. Chem. Phys.*, 125(19):194904, 2006.

[42] S. Peng, F. Ding, B. Urbanc, S. V. Buldyrev, L. Cruz, H. E. Stanley, and N. V. Dokholyan. Discrete molecular dynamics simulations of peptide aggregation. *Phys. Rev. E*, 69(4):041908, 2004.

[43] H. D. Nguyen and C. K. Hall. Molecular Dynamics Simulations of Fibril Formation by Random Coil Peptides. *Proc. Natl. Acad. Sci.*, USA 101:16180, 2004.

[44] A. Donev. Asynchronous event-driven particle algorithms. In *PADS '07: Proceedings of the 21st International Workshop on Principles of Advanced and Distributed Simulation*, pages 83–92, Washington, DC, USA, 2007. IEEE Computer Society.

[45] A. L. Garcia and W. Wagner. Time step truncation error in direct simulation Monte Carlo. *Phys. Fluids*, 12:2621–2633, 2000.

[46] A. L. Garcia and W. Wagner. Generation of the Maxwellian Inflow Distribution. *J. Comput. Phys.*, 217:693–708, 2006.

[47] A. L. Garcia, J. Bell, Wm. Y. Crutchfield, and B. J. Alder. Adaptive Mesh and Algorithm Refinement using Direct Simulation Monte Carlo. *J. Comp. Phys.*, 154:134–155, 1999.

[48] S. Wijesinghe, R. Hornung, A. L. Garcia, and N. Hadjiconstantinou. Three-dimensional Hybrid Continuum-Atomistic Simulations for Multiscale Hydrodynamics. *Journal of Fluids Engineering*, 126:768–777, 2004.

[49] A. Lamura, G. Gompper, T. Ihle, and D. M. Kroll. Multi-particle collision dynamics: Flow around a circular and a square cylinder. *Europhys. Lett.*, 56(3):319–325, 2001.

[50] Y. Inoue, Y. Chen, and H. Ohashi. Development of a Simulation Model for Solid Objects Suspended in a Fluctuating Fluid. *J. Stat. Phys.*, 107(112):85–100, 2002.

[51] F. J. Uribe and A. L. Garcia. Burnett description for plane poiseuille flow. *Phys. Rev. E*, 60(4):4063–4078, 1999.

[52] F. Alexander, A. L. Garcia, and B. J. Alder. Cell Size Dependence of Transport Coefficients in Stochastic Particle Algorithms. *Phys. Fluids*, 10:1540, 1998. Erratum: Phys. Fluids, 12 731 (2000).

[53] T. Bartel, T. Sterk, J. Payne, and B. Preppernau. DSMC simulation of nozzle expansion flow fields. AIAA Paper No. 94-2047, 1994.

[54] S. Gerashchenko and V. Steinberg. Statistics of Tumbling of a Single Polymer Molecule in Shear Flow. *Phys. Rev. Lett.*, 96(3):038304, 2006.

[55] M. A. Novotny. *A Tutorial on Advanced Dynamic Monte Carlo Methods for Systems with Discrete State Spaces*, volume IX of *Annual Reviews of Computational Physics*, pages 153–210. World Scientific, Singapore, 2001.

[56] D. I. Pullin. Direct simulation methods for compressible inviscid ideal-gas flow. *J. Comp. Phys.*, 34:231–244, 1980.

[57] L. Pareschi and S.Trazzi. Numerical solution of the Boltzmann equation by Time Relaxed Monte Carlo (TRMC) methods. *Int. J. Numer. Meth. Fluids*, 48:947–983, 2005.