Computers and Mathematics with Applications 67 (2014) 272–281

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

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

Hybrid molecular–continuum methods: From prototypes to coupling software

Philipp Neumann *, Wolfgang Eckhardt, Hans-Joachim Bungartz

Technische Universität München, Fakultät für Informatik, Boltzmannstr. 3, 85748 Garching, Germany

ARTICLE INFO

Keywords:
Molecular dynamics
Continuum
Lattice Boltzmann
Coupling

ABSTRACT

In this contribution, we review software requirements in hybrid molecular–continuum simulations. For this purpose, we analyze a prototype implementation which combines two frameworks – the Molecular Dynamics framework MarDyn and the framework Peano for spatially adaptive mesh-based simulations – and point out particular challenges of a general coupling software. Based on this analysis, we discuss the software design of our recently published coupling tool. We explain details on its overall structure and show how the challenges that arise in respective couplings are resolved by the software.

1. Introduction

There has been a rising interest in the development and application of molecular–continuum approaches [1] during the last decade. These hybrid schemes allow the investigation of problems which are hardly accessible by either method directly due to their computational intensity or their lack of precision for the given application.

Numerous approaches have been developed by several groups for various particular problem formulations in the context of fluid dynamics. A schematic overview of different methods and related software is provided in Fig. 1. A possible classification separates the methods into particle–particle and particle–mesh coupling methods. Particle–particle approaches couple molecular and continuum methods both of which are based on a Lagrangian fluid description. Examples include DSMC–DSMC (Direct Simulation Monte Carlo) couplings [2,3], couplings of Stochastic Rotation Dynamics (SRT) and Molecular Dynamics (MD) [4,5] or the adaptive resolution scheme (AdResS) [6]. A second branch contains particle–mesh methods in which the molecular motion is resolved by a particle-based method, whereas the continuum flow is modeled by a mesh-based approach such as Navier–Stokes or Lattice Boltzmann simulations. The classification of particle–mesh methods can be further refined into simulations of suspensions [7,8] and multi-level fluid descriptions [9–11]. Various simulation codes and frameworks are available for the simulation of suspensions such as waLBerla [12] or LB3D [13]. For multi-level fluid descriptions, state–state-, flux–state- and flux–flux-based [9,14] schemes have been proposed. One example is given by the coupling method from Dupuis et al. where a Schwarz-like coupling procedure is used to find the steady-state solution of flow around a carbon nanotube [11]. To the authors' knowledge, however, only one framework-like piece of software [15] has been developed so far for the multi-level fluid resolution schemes, exploiting a general purpose coupling framework for the flexible deployment of a sequential molecular–continuum code on grid architectures. Yet, coupling software integrating a continuum with a Molecular Dynamics solver is written from scratch each time, and no standard coupling framework exists. Hence, there are barely papers that address the specific software aspects in respective coupling schemes in order to couple arbitrary mesh-based continuum solvers and Molecular Dynamics implementations. In the literature, only comments on the performance of coupling components can be found, which is not necessarily helpful for the development of a new coupling strategy, as these may apply only to the specific implementations.

* Corresponding author.
E-mail address: neumanph@in.tum.de (P. Neumann).

0898-1221/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.camwa.2013.07.006

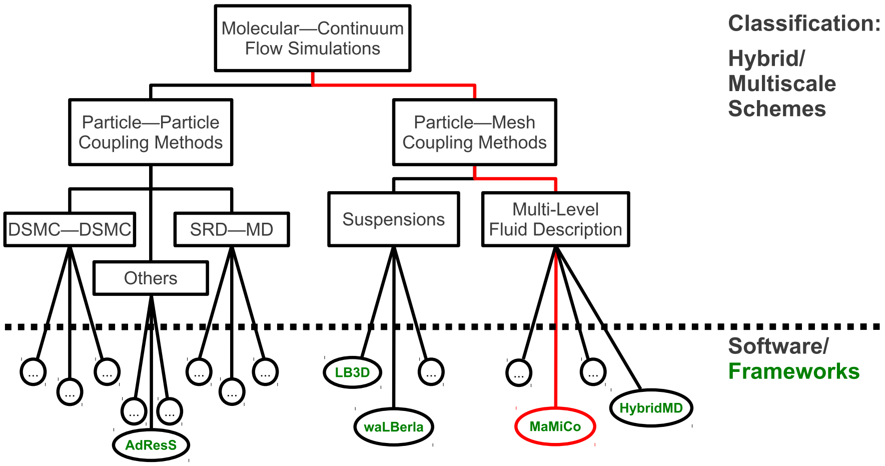

Molecular—Continuum Flow Simulations
Particle—Particle Coupling Methods
Particle—Mesh Coupling Methods
DSMC—DSMC
SRD—MD
Others
Suspensions
Multi-Level Fluid Description
Classification:
Hybrid/ Multiscale Schemes
Software/ Frameworks
...
AdResS
LB3D
waLBerla
MaMiCo
HybridMD

Fig. 1. Classification and available software for molecular–continuum flow simulations. Green-colored pieces of software represent framework-like developments. The red branch represents the classification for the macro–micro-coupling tool [16]. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The purpose of this paper is to highlight the design space of a coupling tool and present a possible general solution which provides all coupling components for mesh–particle molecular–continuum simulations (mostly) independent from the continuum and MD solvers. We recently published aspects in the development of the macro–micro-coupling tool (MaMiCo) toward massively parallel simulations, as well as a brief overview on its functional components [16]. In the following, we want to particularly focus on software aspects and challenges encountered during the prototype formulation.

The paper is organized as follows: first, we give a short review of Molecular Dynamics in Section 2 and introduce mesh-based methods in the context of molecular–continuum coupling schemes in Section 3. We then present a prototype for coupling Molecular Dynamics and a Lattice Boltzmann (LB) automaton in Section 4. Here, we focus on respective general challenges that we encountered during the development of the coupling. We further show that similar challenges may be encountered with any MD code and hence render the general formulation of a respective coupling tool a non-trivial task. Then, we show an approach how to standardize the most common coupling steps. In this context, we review the software layout of our macro–micro-coupling tool (MaMiCo).

2. Molecular dynamics in a nutshell

In MD simulations, a fluid is modeled as a system of N discrete particles. Depending on the purpose of a simulation, different models of rigid-body or flexible molecules have been developed (see e.g. [17]). For the sake of simplicity, however, we describe only the simplest one.

Particles are modeled as rigid spheres, where all particles interact in pairs through a potential $U(r)$. Consequently, the potential energy in the simulation can be calculated as

$$U = \sum_{i=1}^{N} U_i = \sum_{i=1}^{N} \sum_{j \neq i} \frac{1}{2} U(r_{ij}),$$

where r_{ij} denotes the spatial distance of two particles i and j. The total force effective on a molecule i is calculated as the negative gradient of the potential, i.e.

$$F_i = \sum_{j \neq i}^{N} F(r_{ij}) = -\nabla U(r_{ij}).$$

A commonly used potential to describe the interaction of non-polar atoms is the Lennard-Jones-12-6 potential

$$U\left(r_{nm}\right) = 4\epsilon \cdot \left(\left(\frac{\sigma}{r_{nm}} \right)^{12} - \left(\frac{\sigma}{r_{nm}} \right)^{6} \right),$$

where ϵ and σ are material parameters.

The particles are assumed to behave according to Newtonian mechanics, so the time integration of the respective equations of motion can be performed with Verlet-type integrators, e.g. [18]. After the velocity has been updated in each time step, a certain temperature can be imposed on the particles, for example via velocity scaling.

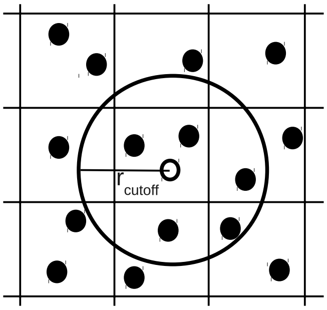


Fig. 2. Schematic of the linked-cells idea.

As the force calculation for one particle involves all others, the resulting algorithm has complexity $O(N^2)$. For short-range potentials such as the Lennard-Jones-12-6, the contributions of distant particles can be neglected or approximated with cut-off corrections, so only particles within a cut-off radius r_c have to be considered. This search for neighboring particles can be efficiently implemented with the Linked-Cells algorithm where the computational domain is subdivided into cells of edge length $l = r_c$ as depicted in Fig. 2. In order to find the neighbors of a given molecule in 3D, only the cell of the molecule and the 26 adjacent cells have to be searched, resulting in a linear complexity of the force calculation.

For the parallelization, a common approach is to apply a domain decomposition scheme, where each process is assigned an equally sized sub-domain, surrounded by a layer of halo cells.

3. Continuum solvers

In contrast to purely particle-based hybrid multiscale schemes such as couplings of dissipative particle dynamics and Molecular Dynamics [6], we consider hybrids which consist of MD and *mesh-based continuum* or *mesoscopic* solvers. Examples for the latter comprise standard (in-)compressible Navier–Stokes solvers or Lattice Boltzmann automata. In mesh-based simulations, the computational domain that shall be treated by the respective solvers is discretized and subdivided in cells. In each cell or vertex of the mesh, the physical quantities, that is mass, momentum or energy, are evaluated by the respective simulation technique. These quantities typically refer to spatial and temporal averages. The continuum solvers hence represent a *coarse-grained* description of the flow system; purely molecular quantities are not captured by them.

4. Prototype development: requirements to the coupling software

Our prototype implementations are based on MarDyn, a framework for massively parallel MD simulations [19,20], and the LB application of the Peano framework [21]. Both are written in C++ and based on well-established software-engineering patterns. The coupling methodology for the hybrid MD–LB simulation is described in [11,22] and is depicted in Fig. 3. The respective coupling algorithm is based on the exchange of state variables and therefore is applicable to steady-state problems, in contrast to flux-based approaches which provide means for time-dependent hybrid molecular–continuum simulations; cf. [9,10]. The major coupling steps common to all hybrid molecular–continuum approaches can be summarized as follows:

1. Solve continuum flow imposing averaged flow quantities from the MD solver. For the scheme in Fig. 3, this resembles several iteration steps of the Lattice Boltzmann automaton until the steady state is reached. In each iteration, the flow velocities sampled in the MD simulation over the red-colored cells are imposed onto the LB system via respective force terms.
2. Send continuum flow quantities from LB to the MD solver. Depending on the underlying coupling methodology, this corresponds to send–receive operations for state or flux variables. In Fig. 3, the flow velocities of the green-colored LB cells are sent to MD which is illustrated by green arrows.
3. Solve MD system imposing the flow quantities from the continuum solver and sample continuum flow quantities. In the present example from Fig. 3, the flow velocities from the green-colored LB cells are imposed onto the molecules within this overlap region over thousands of MD time steps. The average flow velocities are sampled in the red-colored region after reaching equilibrium.
4. Send the averaged flow quantities from MD to the continuum solver. In Fig. 3, the transfer of the average flow velocities from MD to the Lattice Boltzmann solver is visualized by red arrows.

Details of the algorithm shall be reviewed together with respective implications on the interface functionality of the coupling software in the following. As the coupling software should generally be applicable to different kinds of coupling schemes, notes on the requirements of flux-based approaches are given where respective differences between the approaches need to be highlighted. For a detailed description of the realization of the coupling steps 1–4 with the macro–micro-coupling tool, see [16].

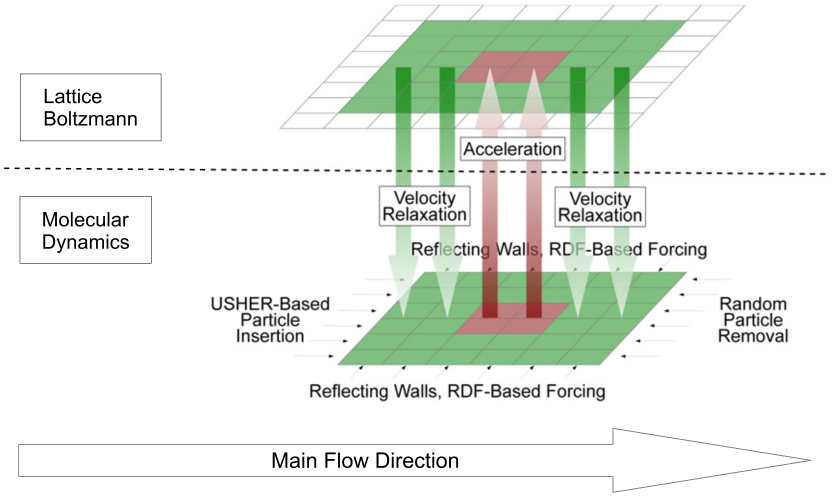


Fig. 3. State-based coupling based on the descriptions in [11]. The overlap region is shown by green-colored cells. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.1. *MD–LB coupling*

First, the coupling from MD to LB is addressed. Within a given control volume which typically corresponds to one LB cell, the macro- or mesoscopic quantities of interest are sampled from the MD simulation. For the current coupling method, this implies averaging in space and time over the velocities of the molecules and sending this averaged velocity to the LB simulation. On the LB side, the averaged velocity is used to determine a forcing term which drives the respective LB cell toward the velocity prescribed by MD. In terms of interfaces between the MD and the LB solver, this means that the coupling implementation requires to hold functionality to trigger sampling of averaged quantities from the MD simulation within a given volume. The same holds for flux-based approaches: in this case, the sampling routine needs to be capable of measuring the difference between subsequent molecular states within a given volume. Storing the old state and defining the difference from the new state hence results in the same requirements. On the LB side, an interface needs to be provided for receiving the quantities from MD. Referring to Fig. 3, only quantities from the inner, that is red-colored, region need to be received. The kind of quantities to be exchanged, for example flux or state variables, is dictated by the coupling scheme. In the following, we restrict the discussions to isothermal flows; energy exchange is to be incorporated into the coupling software analogously to the other quantities in future. From the technical point of view, this does not pose severe differences to the exchange of momentum. Besides the pure receive-operations, an interpretation of the received values is required. This interpretation, however, strongly depends on the coupling method. Considering dimensionless scaling in the MD simulation, the mass or momentum values may need to be re-scaled to LB units or to the respective continuum solver units in the more general case. In order to stay flexible, a respective interface is required. The last step in the MD-to-LB coupling is defined by imposing the interpreted quantities to the LB solver. This part of the coupling is strongly dependent on the coupling method *and* the kind of macro- or mesoscopic solver used in the hybrid method.

Further note that thermal fluctuations may play a crucial role. A consistent incorporation of the fluctuations into the continuum solver may thus be important. Various methods have been proposed to correctly introduce fluctuations for continuum mesh-based solvers such as Navier–Stokes [23] or Lattice Boltzmann solvers [24]. The latter is incorporated into the LB solver of the Peano framework. Here, however, we follow the same approach as Dupuis et al. [11] to be able to compare our results to theirs: we neglect fluctuations on the continuum side and use the standard BGK collision model instead.

4.2. *Coupling LB–MD*

Next, coupling from LB to MD shall be discussed. The transfer of mass, momentum and energy to the molecular system is generally the more challenging step in hybrid simulations.

In the current algorithm, mass and momentum are exchanged within a boundary stripe, that is the green-colored overlap region in Fig. 3. Consider the mass exchange first. Adding a particular amount of mass to a particle system implies the insertion or removal of atoms; access to the memory of the MD simulation is hence necessary. For rarefied molecular systems, a simple random insertion strategy can be sufficient to determine a valid position for a new molecule. For dense fluids, a more sophisticated algorithm is required to find a suitable position for the molecule, avoiding overlaps with other atoms. Most codes therefore make use of the USHER scheme for single-centered [25] or multi-centered atoms [26]: the atoms are initially inserted randomly. Then they are moved in a steepest-descent-like manner on the potential energy landscape described by the existing atoms, until a valid position with respect to the molecule's energy and force is reached or a prescribed maximum number of iteration steps is exceeded. In the case of the latter, the search is restarted with a new random initial position. A controlled step size (and the rotation angle in the case of multi-centered molecules) avoids any unphysical overlap situations with existing atoms during this search. In our prototype, we make use of the USHER scheme for single-centered molecules.

From the USHER description, several interfaces can be formulated for the coupling software. In order to have access to the molecule's quantities, for example position, velocity or force vectors, a wrapper for the molecule description of the underlying MD code is required. Besides, the force acting on a newly inserted atom due to all existing atoms needs to be evaluated. This step can become arbitrarily complex, depending on the intermolecular potentials, or additional forcing terms. A particular problem arises in the case of parallel computations where molecule overlaps can occur, due to the simultaneous insertion of molecules in neighbored regions. A detailed discussion on how to resolve this issue can be found in [16].

The transfer of momentum from the LB to the MD simulation can be established in different ways. We apply a relaxation of the average fluid velocity, i.e. the velocity computed from an average over all molecules within the control volume, toward the target velocity from the LB solver [11]. Other approaches incorporate an additional forcing term which is applied to all relevant molecules; see amongst others [27]. The concept, however, is the same: given the transferred momentum or velocity from the LB simulation, the molecular properties within a given control volume are modified. This modification needs to be carried out such that the changes in molecular forces or velocities are properly taken into account during the time integration of the molecules. Interfaces required hence comprise again the molecule wrapper and an iterator over all molecules inside the control volume. The issue how to define the control volume will be discussed in Section 5.

Though restricted to the isothermal case, we want to shortly comment on energy transfer from meso- or macroscopic solvers to Molecular Dynamics. As temperature is defined via the fluctuations in the molecular velocities, the modification of the molecular thermal energy can be addressed in the same way as the momentum transfer, i.e. via directly modifying the molecular velocities. Consequently, the definition of the required interfaces is very similar as in the case of momentum exchange. In constant temperature simulations, a thermostat has to keep the molecular velocities and hence temperature at the correct level. Common MD codes typically come with thermostats. However, particular thermostats may be required in the hybrid case. Referring to the present coupling scheme [11], the velocities pointing into the main flow directions are not scaled by the thermostat. Whether to reuse existing thermostats or to provide own coupling-specific implementations poses a question to be dealt with during the development of the coupling software.

For all of the aforementioned exchange operations for mass, momentum and energy, the change of one quantity may imply modifications of the other quantities as well. For example, inserting a new molecule does not only add mass to the system but also adds a certain momentum and energy. Hence, a control mechanism is required to retain all quantities except for the one that shall be modified; cf. [16].

4.3. Boundary treatment

The final step in the two-way coupling of Molecular Dynamics with the Lattice Boltzmann solver consists in modeling the boundaries of the open molecular system. As no periodic conditions can typically be applied for these Molecular Dynamics simulations, the potential landscape close to the boundaries needs to be modeled, resulting in additional forcing terms for the molecules that are located close to these boundaries. The underlying models reported so far comprise constant forcing [28], approximations using weighted formulations that depend on the distance of the molecule from the boundary [10,27] and methods which apply radial distribution functions (RDFs) and additional control mechanisms [22,29]. In the approach of Dupuis et al., the RDF-based approach is used: the boundary force is determined via integrating the RDF-weighted molecular interaction force over the molecular interaction volume that is located outside the computational molecular domain. Though Werder et al. illustrated that the RDF-based method shows a severe improvement in the molecular density profile close to the open boundaries, we are not aware of any specific tests that have been previously carried out to validate the behavior of the molecular system close to the boundaries. We further note that the RDF-based approach – similar to the other approaches listed above – does not account for thermal fluctuations. Recent studies have shown that incorporating fluctuations into boundary forcing terms are not straightforward. However, we found that molecular structures are affected in a small region spanning approx. one linked cell layer only but not at bigger distances [30]. Still, the macroscopic behavior can be captured correctly by the respective method, as well as the flow field that arises from the hybrid simulation, compared to reference Molecular Dynamics simulations. The type of boundary force model dictates the interfaces that are required from the coupling software's side. For the boundary models discussed above, a force accumulator per molecule site, similar to the one that was described in the context of USHER-based particle insertions, is sufficient. As additional parameters, the location and size of the molecular domain need to be incorporated into the interface so that the relative position of the molecule to the domain boundaries can be determined. Considering the RDF-based approach, also the type of molecular interaction forces and related parameters need to be known by the interface, such as the Lennard-Jones potential with its characteristic length and potential depth in the present coupling scheme.

More sophisticated boundary models may be required to further reduce the modeling errors close to the Molecular Dynamics–Lattice Boltzmann interface. For example, taking into account the local number density to incorporate local density variations may be important in the case of compressible simulations. Then, a simple interface definition as the one presented here may not be sufficient anymore. Moreover, many ingredients required for boundary force models such as the RDF-based model are already provided by the underlying Molecular Dynamics simulations. For example, sophisticated MD codes such as LAMMPS, NAMD or Amber provide means to sample and output the radial distribution function. A common reader and evaluator for different RDF file formats might therefore be desirable for this type of boundary model. Considering

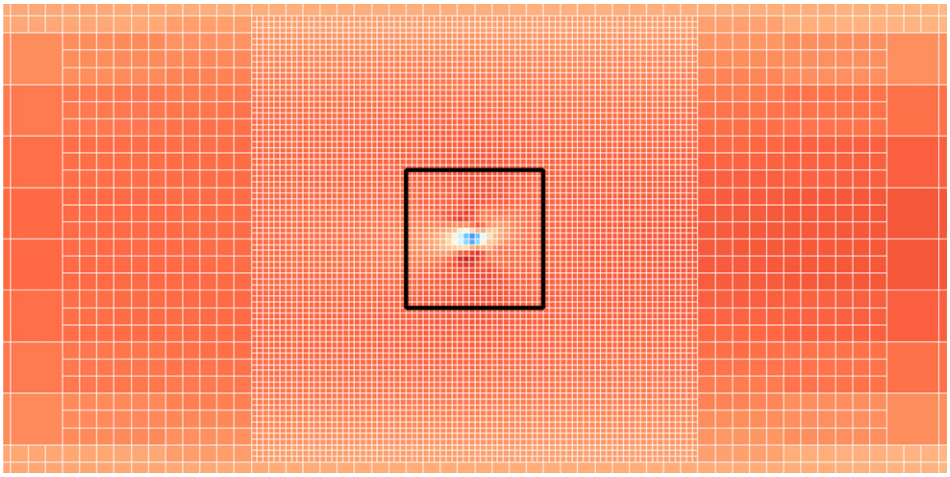


Fig. 4. The velocity field in a coupled Mardyn–Peano simulation of channel flow (Re = 70, Pe = 68) around a small-sized carbon-nanotube. The domain surrounded by thick black lines is solved by Molecular Dynamics; the rest of the domain is solved by the Lattice Boltzmann method, applying a three-level spatially adaptive grid.

all these points, the question arises how much functionality should be plugged in via the coupling software and how much implementational work shall be left to the specific MD software.

Based on the descriptions from above and in [11], a coupling was established combining MarDyn and the spatially adaptive Lattice Boltzmann implementation within Peano. The flow around a carbon-nanotube is shown in Fig. 4: the region close to the nanotube is solved by Molecular Dynamics, whereas the rest of the flow field is computed by the adaptive Lattice Boltzmann scheme.

5. Stepping toward MaMiCo: Software decisions

Having reviewed the major ingredients that are necessary to set up a hybrid molecular–continuum simulation in the previous section, several questions with respect to the required coupling software arose. In the following, possible solutions are sketched, forming the overall concept of the macro–micro-coupling tool.

One open point consists in the choice of the control volume over which molecular quantities such as average velocities shall be sampled. Given the computational domain of the MD simulation, the respective functionality could be realized by defining volumetric iterators: given a sub-domain $[a_0; b_0] \times [a_1; b_1] \times \cdots [a_D; b_D]$, the iterator loops over all molecules and operates only on the molecules within this sub-domain. A similar approach consists in extracting the molecules from the sub-domain and returning an iterator to the new list of molecules. In both cases, at least one traversal over all molecules is necessary. Besides, in massively parallel simulations, additional synchronization between processes is required, since the sub-domain may be chosen arbitrarily and may easily span over several processes. Based on the linked cell approach, a grid structure is already prescribed in the Molecular Dynamics simulations. An iteration scheme over all molecules within one linked cell hence is already available. Depending on the cut-off radius, molecule types, thermodynamic and flow conditions under consideration as well as the coupling scheme itself, a single linked cell may not be of a suitable size for sampling. Therefore, we suggest to reuse the linked cell structure, embedding N^D linked cells into one *macroscopic cell*, with $N \geq 1$; cf. Fig. 5 on the left. In the coupling tool, an interface `MoleculeIterator` is defined for this purpose. Providing an implementation of this interface, an iteration over all molecules within a linked cell becomes possible. The iteration over all linked cells is established inside the coupling tool. No further implementations are required.

Addressing the question on the usage of thermostats, implementations specific to the coupling scheme may be required. In order to keep this step of the coupling hidden from the particular Molecular Dynamics code, we abstained from reusing existing thermostats on the MD side. Instead, separate classes exist within the coupling tool, allowing to control and set the temperature in each macroscopic cell. An exchange of the underlying thermostat hence becomes easily possible.

The interfaces that were found to be required in the previous section are shown in Fig. 5 on the right. Three singletons are introduced: the `CouplingMacroscopicSolverService` and the `CouplingMDSolverService` which are initialized by the respective interface implementations (`MDSolverInterface`, `MacroscopicSolverInterface`) as well as the `MacroscopicCellService` which provides access to the macroscopic cell structure of the coupling tool.

In the following, we discuss the interfaces on the MD side. Implementing the `MoleculeWrapper` allows the coupling tool to access the molecular quantities such as position, velocity, force and energy of a molecule. An implementation of the `MoleculeIterator` yields access to the iterator over all molecules within a linked cell.

For the particle insertion or removal, another interface was identified from the descriptions in the previous section which should allow for allocation or deletion of molecules from the memory of the MD simulation. For this purpose, the interface `MDSolverInterface` provides the methods `addMoleculeToMDSimulation(molecule)` and

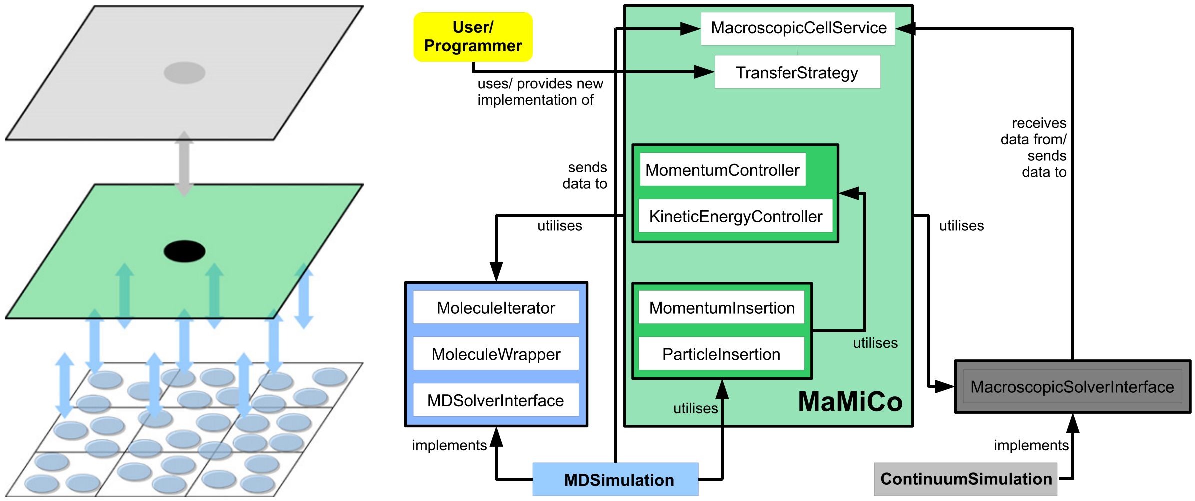


Fig. 5. MaMiCo's principles as published previously by this group [16]. Left: macroscopic cell concept of the coupling tool. N^D linked cells which contain blue-colored molecules are embedded into one green-colored macroscopic cell. Macroscopic quantities are exchanged between the macroscopic cell and the MD simulation or the macroscopic cell and the gray-colored cell of the macroscopic solver, respectively. Right: main pillars of the coupling tool design. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

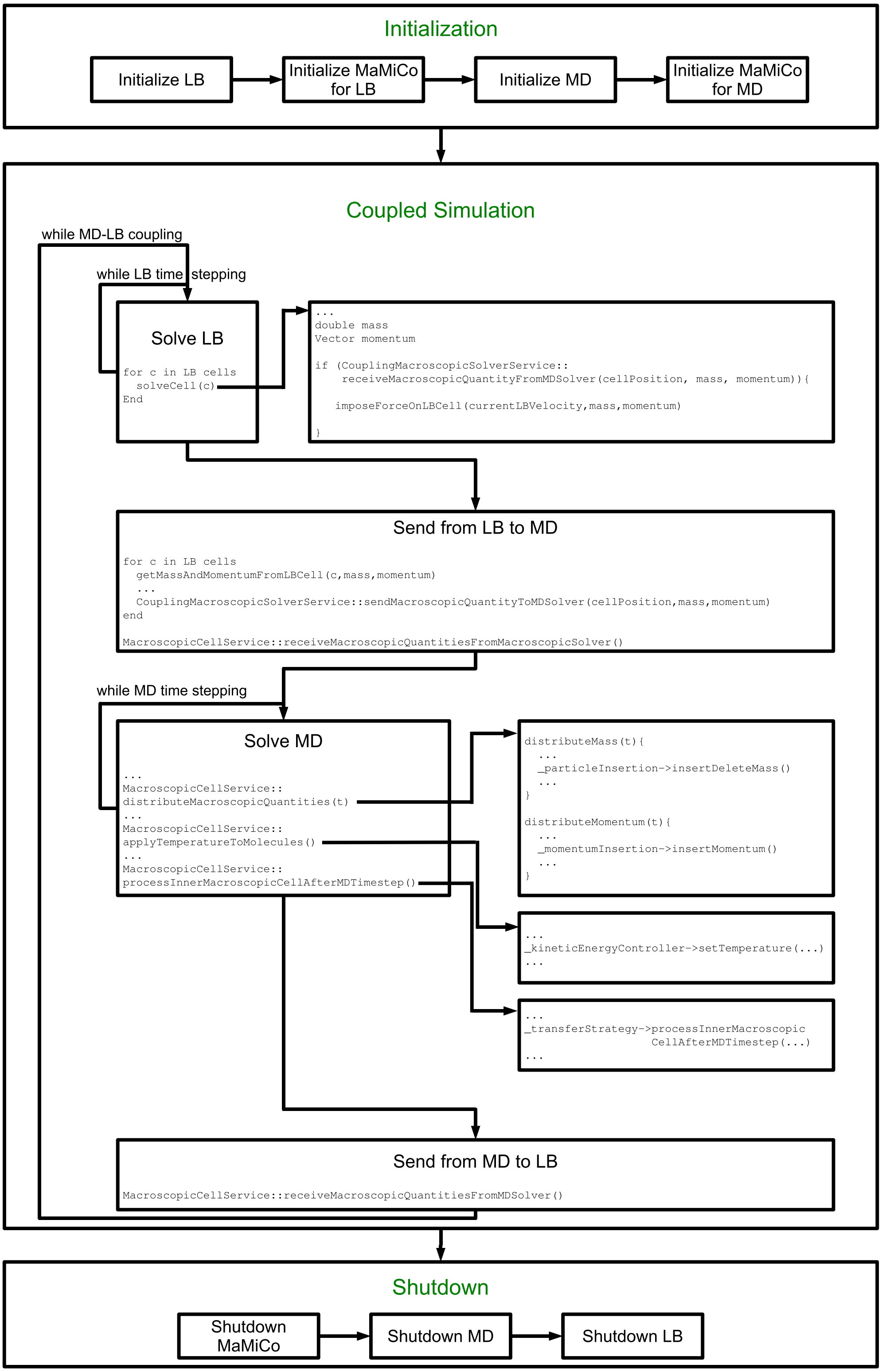
`deleteMoleculeFromMDSimulation(molecule, linkedCell)`. In the case of USHER-based particle insertions, force and energy of the newly inserted molecule need to be extracted from the MD simulation. Thus, the `MDSolverInterface` contains a respective method `computeForceAndEnergy(Molecule,position)`. Having a general method that accumulates all force and energy contributions for this molecule, the interface is kept minimal and leaves maximum freedom to the MD solver in terms of applying different intermolecular potentials or potential fields. On the MD solver side, the interface implementation should be straightforward since this kind of operations needs to be already carried out during the force evaluation in the time integration scheme. The same holds for the setup of the potential energy landscape: given a range of linked cells, this landscape needs to be constructed so that the USHER algorithm can operate on it. Therefore, the method `setupPotentialEnergyLandscape(indexOfFirstLinkedCell,rangeLinkedCells)` is provided by the `MDSolverInterface`.

The insertion mechanisms on the MD solver side – velocity relaxation and USHER-based particle insertion in the case of our prototype – may differ between different coupling schemes. Both `ParticleInsertion` and `MomentumInsertion` are therefore encapsulated in different modules. New insertion mechanisms can be implemented by the application developer, extending the functionality of the coupling tool. The extension and exchangeability of these mechanisms are hence straightforward.

For the exchange of quantities between the solvers, three steps have been identified in Section 4: send quantities, receive them, and interpret them on the continuum solver side. The macroscopic cells introduced for the purpose of sampling are used for the send–receive operations. Encapsulated within the `MacroscopicCellService`, they provide buffers for the exchanged quantities coming from both MD and continuum solver and are always located on the process which holds the embedded linked cells of the MD simulation [16]. Currently, an arbitrary domain decomposition for the continuum solver and a regular blockwise domain decomposition for the MD solver is assumed; the latter may be extended in future to a more general case. The `MacroscopicSolverInterface` provides methods to set up the respective parallel topology between the coupling tool and the continuum solver. The `MacroscopicCellService` provides methods for the send-/receive-operations between the continuum solver and the coupling tool as well as the MD solver and the coupling tool, respectively.

The interpretation of the quantities that are stored in the macroscopic cells can be achieved via implementing a `TransferStrategy`. Operations on the different macroscopic cell buffers such as unit conversions before the transfer of the quantities to the continuum solver or sampling values after each MD time step can be implemented in a cellwise manner within a separate `TransferStrategy` for each new coupling scheme. Besides, the `TransferStrategy` can steer in which cells macroscopic quantities shall be received/sent from/to the LB solver, allowing to define the underlying atomistic–continuum domain decomposition, overlap regions, etc. They are triggered via a callback-mechanism for each macroscopic cell from the `MacroscopicCellService`. For example, the method `processInnerMacroscopicCellAfterMDTimestep(cell,index)` from the `TransferStrategy` is implemented in order to sample data in the MD simulation. Calling `processInnerMacroscopicCellAfterMDTimestep()` of the `MacroscopicCellService` after each MD time step triggers the sampling procedure on all macroscopic cells.

Finally, imposing the quantities to the continuum solver is established via the `CouplingMacroscopicSolverService`: getters and setters (for a certain grid cell at position x in space and with a meshsize h) are available from this singleton.



```
for c in LB cells
  solveCell(c)
End
```

```
...
double mass
Vector momentum

if (CouplingMacroscopicSolverService::
    receiveMacroscopicQuantityFromMDSolver(cellPosition, mass, momentum)){

   imposeForceOnLBCell(currentLBVelocity,mass,momentum)

}
```

```
for c in LB cells
  getMassAndMomentumFromLBCell(c,mass,momentum)
  ...
  CouplingMacroscopicSolverService::sendMacroscopicQuantityToMDSolver(cellPosition,mass,momentum)
end

MacroscopicCellService::receiveMacroscopicQuantitiesFromMacroscopicSolver()
```

```
...
MacroscopicCellService::
distributeMacroscopicQuantities(t)
...
MacroscopicCellService::
applyTemperatureToMolecules()
...
MacroscopicCellService::
processInnerMacroscopicCellAfterMDTimestep()
```

```
distributeMass(t){
  ...
  _particleInsertion->insertDeleteMass()
  ...
}

distributeMomentum(t){
  ...
  _momentumInsertion->insertMomentum()
  ...
}
```

```
...
_kineticEnergyController->setTemperature(...)
...
```

```
...
_transferStrategy->processInnerMacroscopic
                    CellAfterMDTimestep(...)
...
```

```
MacroscopicCellService::receiveMacroscopicQuantitiesFromMDSolver()
```

Fig. 6. Flow of a coupled molecular–continuum simulation using the coupling tool.

Putting all aspects together, the flow of a coupled simulation is shown in Fig. 6: the send–receive operations are triggered via calls to the respective services. Besides, each component – moment modification components (such as particle or momentum insertion), controller mechanisms (such as the KineticEnergyController used as thermostat) and the

TransferStrategy – is clearly separated and thus independent from the underlying LB and MD simulation and all other components.

6. Conclusion

We discussed in depth the software design of our coupling tool. We pointed out major challenges in the development, and how to address those, such as the compatibility to other MD and continuum codes or how to keep the components of the coupling encapsulated and easily exchangeable (velocity relaxation, particle insertion). We further discussed how to implement different coupling and quantity transfer strategies while still preserving a general coupling implementation. We therefore described the design with respect to the TransferStrategy, the ParticleInsertion and MomentumInsertion modules. Still, the generality of this approach comes at the cost of additional controller mechanisms to allow for the modification of either mass or momentum. The discussion was restricted to isothermal simulations so far. The inclusion of energy transport is to be incorporated in future. One question that arose in the Section 4 was the incorporation of the open boundary treatment on the MD solver side. From the authors' point of view, a general purpose open boundary model for molecular systems is – despite the numerous models proposed so far – still an open research topic. This particularly holds for the boundary treatment for complex, e.g. multi-centered Lennard-Jones, liquids where to the authors' knowledge only the model by Kotsalis et al. [29] incorporates local molecular structures. For this reason, the open boundary treatment is currently left to the MD solver. More work in this direction will follow and is subject to current developments. The accomplishment of a totally three-dimensional adaptive MD–LB scheme using the coupling tool is to be published soon. In order to prove our concepts to be generally applicable, coupling of arbitrary molecular and continuum solvers such as LAMMPS, ESPResSo or Palabos, waLBerla using the macro–micro-coupling tool is required. Further software and code analysis in this direction is part of future work.

Acknowledgments

We gratefully acknowledge the support of the Faculty Graduate Center CeDoSIA at the Technische Universität München. We further thank the King Abdullah University of Science and Technology for providing computational resources (project K117) and Peter Hoffmann for his work on the prototype development.

References

[1] H.S. Wijesinghe, N.G. Hadjiconstantinou, Hybrid atomistic—continuum formulations for multiscale hydrodynamics, in: Handbook of Materials Modeling, Springer Science + Business Media B.V., 2005, pp. 2523–2551.
[2] J.M. Burt, I.D. Boyd, A hybrid particle approach for continuum and rarefied flow simulation, J. Comput. Phys. 228 (2008) 460–475.
[3] A. Donev, B.J. Alder, A.L. Garcia, A thermodynamically-consistent non-ideal stochastic hard-sphere fluid, J. Stat. Mech. 11 (2009) P11008.
[4] M. Hecht, Simulation of Peloids, Dissertation, 2007.
[5] G. Gompper, T. Ihle, D.M. Kroll, R.G. Winkler, Multi-particle collision dynamics—a particle-based mescoscale simulation approach to the hydrodynamics of complex fluids, Adv. Polym. Sci. 221 (2009) 89.
[6] R. Delgado-Buscalioni, K. Kremer, M. Praprotnik, Concurrent triple-scale simulation of molecular liquids, J. Chem. Phys. 128 (2008) 114110.
[7] C.K. Aidun, J.R. Clausen, Lattice-Boltzmann method for complex flows, Annu. Rev. Fluid Mech. 42 (2010) 439–472.
[8] B. Dünweg, A.J.C. Ladd, Lattice Boltzmann simulations of soft matter systems, Adv. Polym. Sci. 221 (2009) 89.
[9] W. Ren, Analytical and numerical study of coupled atomistic–continuum methods for fluids, J. Comput. Phys. 227 (2007) 1353–1371.
[10] R. Delgado-Buscalioni, P.V. Coveney, Continuum–particle hybrid coupling for mass, momentum and energy transfers in unsteady fluid flow, Phys. Rev. E 67 (2003) 046704.
[11] A. Dupuis, E.M. Kotsalis, P. Koumoutsakos, Coupling lattice Boltzmann and molecular dynamics models for dense fluids, Phys. Rev. E 75 (2007) 046704.
[12] C. Feichtinger, J. Götz, S. Donath, K. Iglberger, U. Rüde, WaLBerla: exploiting massively parallel systems for lattice Boltzmann simulations, Parallel Comput. (2009) 241–260.
[13] D. Groen, O. Henrich, F. Janoschek, P.V. Coveney, J. Harting, Lattice-Boltzmann methods in fluid dynamics: turbulence and complex colloidal fluids, in: Juelich Blue Gene/P Extreme Scaling Workshop 2011, Juelich Supercomputing Centre, 2011. http://ccpforge.cse.rl.ac.uk/gf/project/lb3d/.
[14] T.H. Yen, C.Y. Soong, P.Y. Tzeng, Hybrid molecular dynamics—continuum simulation for nano/mesoscale channel flows, Microfluid. Nanofluid. 3 (2007) 665–675.
[15] R. Delgado-Buscalioni, P.V. Coveney, G.D. Riley, R.W. Ford, Hybrid molecular-continuum fluid models: implementation within a general coupling framework, Phil. Trans. R. Soc. A (2005).
[16] P. Neumann, N. Tchipev, A coupling tool for parallel molecular dynamics—continuum simulations, in: Published in Proceedings of the International Symposium on Distributed and Parallel Computing 2012, 2012.
[17] D.C. Rapaport, The Art of Molecular Dynamics Simulation, Cambridge University Press, 1996.
[18] L. Verlet, Computer Experiments on Classical Fluids, Phys. Rev. 159 (1967) 98.
[19] M. Buchholz, H.-J. Bungartz, J. Vrabec, Software design for a highly parallel molecular dynamics simulation framework in chemical engineering, J. Comput. Sci. 2 (2) (2011) 124–129.
[20] C. Niethammer, M. Buchholz, M. Bernreuther, M. Horsch, W. Eckhardt, H. Hasse, H.-J. Bungartz, J. Vrabec, ls1 mardyn: a massively parallel molecular dynamics simulation tool for applications in chemical engineering, in preparation.
[21] P. Neumann, T. Neckel, A dynamic mesh refinement technique for lattice Boltzmann simulations on octree-like grids, Comput. Mech. (2012) http://dx.doi.org/10.1007/s00466-012-0721-y. Published online.
[22] T. Werder, J.H. Walther, P. Koumoutsakos, Hybrid atomistic–continuum method for the simulation of dense fluid flows, J. Comput. Phys. 205 (2005) 373–390.
[23] A. Donev, E. Vanden-Eijnden, A. Garcia, J. Bell, On the accuracy of explicit finite-volume schemes for fluctuating hydrodynamics, Commun. Appl. Math. Comput. Sci. 5 (2010) 149–197.
[24] B. Dünweg, U.D. Schiller, A.J. Ladd, Statistical mechanics of the fluctuating lattice Boltzmann equation, Phys. Rev. E 76 (2007) 036704.
[25] R. Delgado-Buscalioni, P.V. Coveney, USHER: an algorithm for particle insertion in dense fluids, J. Chem. Phys. 119 (2) (2003) 978–987.

[26] G. De Fabritiis, R. Delgado-Buscalioni, P.V. Coveney, Energy controlled insertion of polar molecules in dense fluids, J. Chem. Phys. 121 (24) (2004) 12139–12142.

[27] E.G. Flekkøy, G. Wagner, J. Feder, Hybrid model for combined particle and continuum dynamics, Europhys. Lett. 52 (3) (2000) 271–276.

[28] S.T. O'Connell, P.A. Thompson, Molecular dynamics—continuum hybrid computations: a tool for studying complex fluid flow, Phys. Rev. E 52 (6) (1995) R5792–R5795.

[29] E.M. Kotsalis, J.H. Walther, E. Kaxiras, P. Koumoutsakos, Control algorithm for multiscale flow simulations of water, Phys. Rev. E 79 (2009) 045701(R).

[30] P. Neumann, W. Eckhardt, H.-J. Bungartz, A radial distribution function-based open boundary force model for multi-centered molecules, submitted for publication.