Integrating Multi-level Molecular Simulations across Heterogeneous Resources

Yudong Sun #1, Steve McKeever #2, Kia Balali-Mood *3, Mark S. P. Sansom *4

Computing Laboratory, University of Oxford
Parks Road, Oxford OX1 3QD, UK
1 Yudong.Sun@comlab.ox.ac.uk
2 Steve.McKeever@comlab.ox.ac.uk

* *Department of Biochemistry, University of Oxford*
South Parks Road, Oxford OX1 3QU, UK
3 Kia.Balali-Mood@bioch.ox.ac.uk
4 Mark.Sansom@bioch.ox.ac.uk

Abstract— **Biomolecular simulations play a key role in the study of complex biological processes at microscopic levels in which macromolecules such as proteins are involved. The simulations are usually computationally demanding and no single method can achieve all levels of details. Thus, the simulations at different levels need to be integrated to jointly manifest atomic insights into these processes. This paper presents a Grid-based simulation framework to support the integration of multi-level simulations by means of dynamic coupling, automated workflow management, resource-dependent job distribution, and XML-based data representation. The framework provides an e-Science infrastructure to support biomolecular simulations on Grids. A biomolecular simulation markup language called BioSimML is developed to provide a formatted data representation to the multi-level simulations. Experimental simulations have shown flexible integration and high performance enhancement achieved in molecular simulations based on our framework.**

I. Introduction

Understanding dynamic interactions of macromolecules such as proteins with other molecules in a cellular environment has pharmaceutical significance in aiding the design of new drugs [1], [2]. These interactions represent the biological processes at microscopic levels spanning from quantum chemical to cell biological level, which are not always tractable by lab experiments. Biomolecular simulations enable us to study and understand these biological processes with atomic details using biochemical and computational methods.

Biomolecular simulations can be undertaken at different levels such as Quantum Mechanics (QM) level, Molecular Mechanics (MM) level, and Molecular Dynamics (MD) level [3]. The simulations on different levels aim to investigate different scientific phenomena. For instance, QM/MM simulations are used to study chemical reactions, whereas MD simulations are used to understand the overall dynamics of a system. Therefore, no single simulation method can capture all the different levels of details. We need to integrate the simulations on these levels to expose all required constituents of the biological processes. Nevertheless, a biomolecular simulation is usually a computationally demanding process with a large number of molecules and atoms as well as complicated interactions involved. There exist various software tools to conduct such simulations, for example, GROMACS [4], NAMD [5], and CHARMM [6] that have varied data representations.

Therefore, dynamic coupling of multi-level simulations is essential to a seamless integration and efficient simulation models are demanded to improve their performance. The simulations need to be executed utilising high performance computing resources such as on a Grid. A formatted data representation is required to formulate data exchange and transformations between different simulation levels and across distributed computing resources. To achieve all these targets, we have developed a simulation framework to implement seamless integration of multi-level molecular simulations based on a Grid. The framework provides a unified infrastructure for the integration of simulations by means of dynamic coupling, automated workflow management, formulated data exchange, and resource-dependent job distribution. A biomolecular simulation markup language, BioSimML, has been designed to provide an XML-based document structure to formulate the data exchange and transformations between multi-level simulations.

The framework has established an e-Science platform on the UK National Grid Service NGS (www.ngs.ac.uk) to realise multi-level biomolecular simulations for the research in biochemistry and computational systems biology. Experimental simulations including a multiscale MD simulation model that integrates the simulations at atomistic and coarse-grained scales and the data transformation between MD and QM/MM simulation levels have shown the flexible integration and prominent performance enhancement achieved by our framework.

The rest of the paper is organised as follows. Section II compares our research with related work. Section III presents the multi-level simulation framework. Section IV introduces the BioSimML language. Section V demonstrates two examples implemented based on the framework. Section VI concludes the paper and discusses future work.

II. Related Work

The combination of multi-level molecular simulation models has been a new methodology arising in recent years in biological research. The Integrative Biology project [7] is developing multiscale simulation models of heart diseases and cancer tumours, ranging from genes to whole organs, in order to understand the causes of such diseases. The project also explores the access to Grid computing and database resources to run coupled multiscale whole organ simulations. HybridMD [8], [9] developed a multiscale hybrid model for the simulation of complex fluid flow in a cellular environment, for which two contiguous subdomains are dynamically coupled in a molecular system, in order to reduce the computational overhead. For instance, one subdomain of a protein and surrounding water is modelled by atomistic molecular dynamics and the other subdomain of bulk water is modelled in coarse-grained continuum fluid dynamics. The model is implemented within a general coupling framework (GCF) to enable the deployment on disparate architectures including Grid. BioSimGrid [10] provides a Grid-enabled environment that allows users to share biomolecular simulation data produced in different institutions by different simulation tools and to perform fundamental analysis on the data.

Our simulations concentrate on the behaviour and interaction of macromolecules at microscopic levels. The simulations on distinct levels are dynamically integrated to manifest a full representation of the biological details. Compared with the related work above, our simulation framework is unique in the support for dynamic and flexible integration of multi-level simulations and in the automation of integration workflow that the related projects do not or have not provided. The framework provides not only a conceptual model but also an implementation of runtime integration of biomolecular simulations on a Grid.

XML-based data representations have been used to capture biological models and data in recent years. CellML [11] is an XML-based language for defining mathematical models of cellular functions such as biological pathways and electrophysiological models to facilitate the reuse of the models in biological community. It embeds MathML [12] to represent mathematical equations. SBML [13] is an XML-based format for representing the biochemical reaction networks common to computational biology research, such as cell signalling pathways, metabolic pathways, and gene regulation. The PDB (Protein Data Bank) format [14] provides a standard representation for macromolecular structures, which is widely used by molecular simulation tools. PDBML [15] is the XML document structure for the PDB Exchange dictionary. The PDB Exchange dictionary is a superset of the PDB protein structure data, described in the mmCIF (macromolecular Crystallographic Information File) format which is significantly different from the PDB format. As the designers indicated, it is not possible to completely automate the conversion of a PDB file to an mmCIF one [16].

To formulate the data exchange and transformation in the integration of multi-level simulations, we have developed the BioSimML markup language based on XML constructs to describe molecular simulations and capture molecular structures originally represented in the PDB format. The PDB format is widely used by the molecular simulation tools in our multi-level simulations, however, it cannot be directly captured by PDBML. BioSimML provides the XML document structure to represent the PDB format of molecular structures and help the transformation of data format in our simulations.

III. The Multi-level Simulation Framework

In our research, we study biomolecular processes which are involved in the interactions between membrane-bound proteins and pharmaceutical drugs. The membrane-bound proteins are from pathogens that infect living things. The research will find the proper structures of drugs that can bind to the pathogenic proteins and reduce their activity to inhibit infection. These proteins show complex conformational and dynamic behaviour. Their interactions need to be modelled at different levels. We have been undertaking the simulations on four levels as shown in Fig. 1:

- *Level 1:* QM/MM models the mechanisms of protein reactions.
- *Level 2:* Drug Docking provides candidate structures of drugs docked (bound) at the active sites of the proteins.
- *Level 3:* MD simulates proteins attaching to cell membrane through their interactions.
- *Level 4:* Drug Diffusion studies the permeation and diffusion of potential drugs in cell membrane.

The four simulation levels need to be integrated together to exchange information. One simulation level requires input from another level and the output of one level may feed back to another level. We need to seamlessly integrate the simulation levels to enable their interactions. Fig. 1 shows the integration of the simulation levels. For example, the QM/MM level requires the molecular structures from the drug diffusion level as the starting points for QM/MM simulation. The QM/MM level also exchanges data with the drug docking and MD levels.

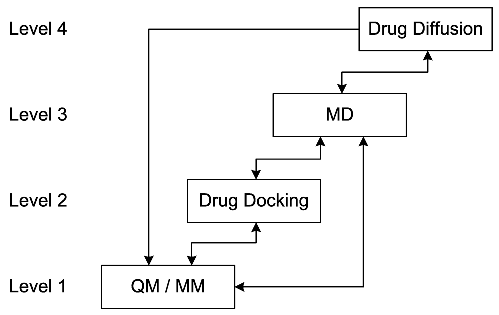


Fig. 1. The interaction of simulation levels.

Flexible integration is required across the simulations levels. The integration can be established at runtime where concurrently running simulations can exchange data or switch from one simulation level to another level at an appropriate point. This runtime integration can be considered as a *tightly-coupled* integration mode. In addition, off-line integration is also needed by which simulations will be individually executed without runtime interaction. Instead, the output of one simulation level will subsequently be forwarded to the next simulation level as the input. This is a *loosely-coupled* integration mode. As biomolecular simulations are computationally demanding on all levels, multi-level simulations are required to run on high-performance computing systems. Various software tools are employed to run the simulations on different levels and to perform data transformations between the levels. We have been using molecular simulation tools GROMACS, NAMD, CHARMM, and TINKER [17] as well as some molecular structure modelling and analysis tools. In order to efficiently integrate the simulations, we need to meet the following requirements:

1) Formulated Data Representation: The simulation tools on different levels use different data representations. The data exchange between simulation levels usually requires transforming data formats. Even the same data format has variations depending on specific simulation tools. Further, the integration of tightly-coupled simulations need to exchange data crossing distributed systems at runtime. The data exchange must be formulated by a unified data representation.

2) Flexible Integration: A simulation on one level needs to be flexibly integrated to a simulation on another level. Such integration requirements will emerge in the progress of our research rather than stemming from a predefined simulation scheme. The integration should be achieved by simulation invocation and data exchange between the levels in either tightly-coupled or loosely-coupled mode.

3) Dynamic Coupling: This requirement is essential to tightly-coupled simulations which need dynamic integration at runtime. The output of a simulation on one level will be pushed forward as the input to a simulation on another level. A simulation on one level can also be switched to a simulation on another level depending on specified conditions occurring in the simulation. We need to develop the mechanism to realise the dynamic coupling. The mechanism will check the state of a simulation to determine a point where data exchange or simulation switch can be conducted and to implement the integration process including data transformation and exchange, and triggering a simulation on another level. The whole workflow of dynamic coupling needs to be automated for an efficient integration. Due to the use of high-performance computing resources, the mechanism needs to support the dynamic coupling of simulations across distributed systems in particular on a Grid.

4) Resource-dependent Job Distribution: The integration of multi-level simulations includes various operations from simulation execution to data transformation. The operations require different hardware and software resources. Computationally demanding simulations need to run on high-performance systems where the required simulation software should be available. Data transformation may use special software that is only installed on a client machine. Hence, the operations in a simulation integration need to be distributed onto heterogeneous systems to use the available resources on demand. In the mean time, these operations need to be coordinated to realise an integrated simulation workflow.

5) Unified Computing Infrastructure: With respect to the heterogeneous resources used in multi-level simulations, a unified computing infrastructure is demanded to effectively integrate the simulations across heterogeneous systems. That is, each simulation is resource-dependent, whereas the entire simulation infrastructure ought to be resource-neutral to allow seamless integration of the simulations on all levels and across heterogeneous resources.

In response to all these requirements, we have developed a simulation framework that provides the fundamental components as well as an execution infrastructure for the integration of multi-level molecular simulations. We have also developed an XML-based markup language BioSimML to capture the data in the simulations, which will be presented in Section IV. Fig. 2 shows the framework created to integrate two simulations running on two computing systems. In the Figure, the framework is constructed with two counterparts. Part A is responsible for running the simulation on level L_A. Part B is responsible for the simulation on level L_B. Each is deployed on one of the systems. A two-level simulation can be established by the interaction between the two parts. The major components in each part are:

1) *Simulation manager* coordinates the entire simulation procedure on one level and interacts with the simulation manager in another part to integrate the simulations between them. It invokes other components to complete the operations required for the integration of the simulations.
2) *Simulation monitor* checks the state of an ongoing simulation in order to derive a time point on which the simulation can exchange data with another simulation or trigger a simulation on another level.
3) *Simulation executor* calls a simulation tool (e.g. GROMACS or NAMD) to run a molecular simulation.
4) *Data converter* transforms the data produced by a simulation to the data format required by another simulation for data exchange.
5) *Data repository* is a collection of input and output data of a simulation.
6) *BioSimML handler* implements an XML-based parser to capture the simulation data in BioSimML. The BioSimML handler on the sender side encodes flat data into a BioSimML document and the handler on the receiver side converts it back.

The component-based framework can integrate multi-level simulations running on distributed systems via the following workflow. Let Part A of the framework initiate a simulation on level L_A under the control of the simulation manager. The simulation manager calls the simulation executor to run the simulation. The input and output data of the simulation are held in the data repository. Meanwhile, the simulation manager activates the simulation monitor to routinely check the progress of the simulation in order to determine a time

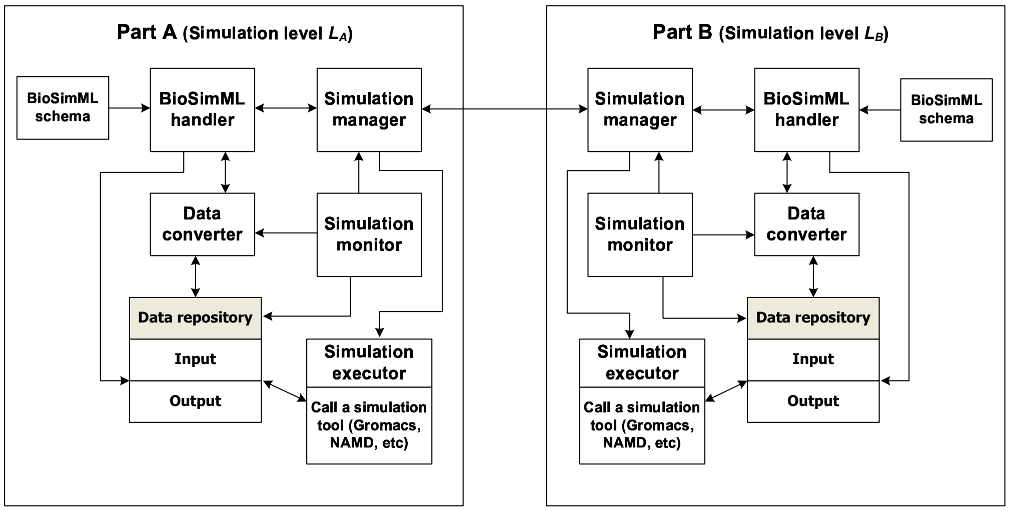


Fig. 2. A multi-level simulation framework

point when the simulation on level L_A can be integrated to a simulation on level L_B. The time point can be called an *integration point* which is derived by a predefined condition occurring in the simulation. When an integration point is achieved, the simulation monitor notifies the simulation manager of this event and calls the data converter to transform the data from its current form to the format required by a simulation on level L_B. After the transformation, the data will be captured into a BioSimML document using the BioSimML handler. Finally, the simulation manager in Part A interacts with the simulation manager in Part B to conduct a simulation integration and sends the BioSimML captured data to Part B as the input data to the simulation on level L_B.

The data transformation can also be manipulated in Part B instead of Part A. In this case, the data produced by the simulation in Part A will be directly captured in BioSimML and transmitted to Part B. The data converter in Part B will then perform the data transformation and feed the data to the local simulation. Furthermore, the data transformation can be collaboratively accomplished by the data converters in both parts. Each converter is responsible for transforming certain type of data. The transformed data in each part will then be assembled together in Part B as the input to the simulation on level L_B. The collaborative data transformation can speed up the transformation process and enable a resource-dependent distributed data transformation method. Section V-A will show the collaborative data transformation in a multiscale MD simulation model.

In Part B, the simulation manager receives the data and delivers it to the BioSimML handler which in turn decodes the data into flat files. Then, the simulation manager triggers the simulation executor to start a simulation on level L_B using the received data and activates the simulation monitor to inspect the simulation progress. Thereupon, a runtime integration of two simulations has been accomplished across the two systems. The simulation in Part A can either continue or cease to work depending on a simulation plan.

The framework can flexibly organise multi-level simulations in different integration models. The simulation on level L_B, for example, can be integrated to a simulation on level L_C. Therefore, the simulations can be pipelined from level L_A to L_C through L_B. The simulation on level L_A can also be integrated to a simulation on level L_D in addition to the integrated simulation on level L_B. This will produce a tree-structured simulation model.

The framework provides a unified infrastructure that is able to support seamless integration of the simulations on different levels and on different computing resources, and to automate the integration workflow. It is implemented as a Java package for cross-platform compatibility. The package will be deployed on the computing systems where multi-level molecular simulations will be run and integrated. The framework is specified as a software skeleton which encapsulates the fundamental components as described above. The skeleton can be implemented and extended by varied classes to accommodate a multitude of simulation methods, tools, hardware and software requirements. The framework is embedded with BioSimML to support formulated data exchange. Section V-A will introduce an integrated multiscale simulation model implemented based on this framework.

IV. BioSimML

Multi-level simulations need a unified data representation to formulate the data transformation and exchange between simulation levels and across heterogeneous systems. We have been developing BioSimML, an XML-based biomolecular simulation markup language. BioSimML provides an ontology to describe simulation metadata, simulation parameters, and molecular structures. A BioSimML document describes a GROMACS-based MD simulation as below:

```
<?xml version="1.0" encoding="UTF-8"?>
<biosimml>
  <!-- Metadata -->
  <description>
```

```
    <level scale="atomistic">Molecular dynamics</level>
    <tool version="3.3.1">Gromacs</tool>
    <mode>parallel</mode>
    <molecule serial="1" type="protein">gp160</molecule>
    <molecule serial="2" type="lipid">dppc</molecule>
    <molecule serial="3" type="water">sol</molecule>
  </description>

  <!-- PDB format -->
  <pdb>
    <headers>
      <title name="HEADER" content="SIV gp160, DPPC, SOL">
    </headers>
    <cryst1 keyword="CRYST1" a="61.250" b="125.000"
            c="95.000" alpha="90.00" beta="90.00"
            gamma="90.00" sgroup="P 1" z="1"/>
    <model serial="1">
      <atom serial="1" atom_name="CA" res_name="GLY"
            res_seq="1" x="10.577" y="76.986" z="16.166"
            occupancy="1.00" tempFactor="0.00"/>
      ......
      <atom serial="119" atom_name="C1" res_name="DPP"
            res_seq="13" x="51.697" y="51.352" z="59.442"
            occupancy="1.00" tempFactor="0.00"/>
      ......
      <atom serial="12919" atom_name="OW" res_name="SOL"
            res_seq="269" x="3.674" y="22.560" z="14.297"
            occupancy="1.00" tempFactor="0.00"/>
      ......
    </model>
  </pdb>

  <!-- mdp options -->
  <mdp>
    <run_control comment="RUN CONTROL PARAMETERS">
      <integrator>md</integrator>
      <comment>start time and timestep in ps</comment>
      <tinit>0.0</tinit>
      <dt>0.002</dt>
      <nsteps>750000</nsteps>
      ......
    </run_control>
    <output_control comment="OUTPUT CONTROL OPTIONS">
      <comment>Output frequency for coords (x),
               velocities (v) and forces (f)</comment>
      <nstxout>1000</nstxout>
      <nstvout>1000</nstvout>
      <nstfout>1000</nstfout>
      ......
    </output_control>
    ......
  </mdp>
</biosimml>
```

The BioSimML document above contains three parts. The `<description>` element is the metadata of a simulation. It describes the simulation level and scale (i.e. molecular dynamics level and atomistic scale), the simulation tool and execution mode (GROMACS 3.3.1, parallel run), and the types of molecule involved (gp160 protein, DPPC lipid, and water).

The second part, `<pdb>` element, is the XML construct we designed to capture the molecular structures originally represented in the PDB format. The inner `<atom>` element describes the ATOM record of the PDB format which defines an atom's name, serial number, Cartesian coordinates, and other properties. The three `<atom>` elements shown in the code above represent an α-carbon atom in a gp160 protein, a carbon atom in a DPPC lipid, and an oxygen atom in a water molecule.

The third part is the `<mdp>` element that captures the mdp (molecular dynamics parameter) file of GROMACS, which defines the conditions of an MD simulation. The `<run_control>` element captures the run control parameters. For example, the `<integrator>` encloses the value `md` which means using Newton's equations of motion to calculate the movement of molecules. The `<tinit>` sets the start time of simulation to 0. The `<dt>` defines the time step as 0.002ps and the `<nsteps>` is the total number of simulation steps 750000. The BioSimML code actually specifies an atomistic MD simulation which will be discussed in Section V-A.

We have developed a BioSimML parser to convert a PDB file and an mdp file into a BioSimML document and vice versa. The BioSimML document structure is specified in RELAX NG syntax [18], which is a standard XML schema language for the specification and validation of XML document structure. The RELAX NG specification enables the validation of BioSimML captured data against the formal specification. Section V-B will show an example of how BioSimML is used for the data transformation between two simulation levels.

V. Examples

This section demonstrates two use cases of the multi-level simulation framework and BioSimML. One is a multiscale MD simulation model developed based on the framework. The other is the use of BioSimML in data transformation between MD and QM/MM levels.

A. A Multiscale MD Simulation Model

The multi-level simulation framework has been used to implement a multiscale simulation model which integrates molecular dynamics simulations at two granularities: *atomistic* and *coarse-grained* scales. An atomistic simulation is based on an all-atom model that computes atom-to-atom interactions [19]. Since a molecular system usually contains tens of thousands of atoms, an atomistic simulation is extremely computationally-complex. To reduce the computational complexity, a coarse-grained model can be used in which a group of bonded atoms are represented by a single particle (usually the centre of mass) and the interactions are only computed between the particles [20]. With a reduced system size, a coarse-grained simulation can advance more rapidly. However, a coarse-grained simulation cannot accurately mimic the atomic details of a biological process.

To achieve a fast simulation progress and meanwhile reveal atomic details, we have developed a multiscale MD simulation model which combines coarse-grained and atomistic simulations into an integrated model. Fig. 3(a) shows the workflow of the multiscale simulation model. The multiscale model starts a simulation at the coarse-grained (CG) scale for a rapid progress to set up the initial molecular system. An integration point is derived based on the equilibrium state of the molecular system. The equilibrium state is assessed by the coordinate root mean square deviation (RMSD) between the molecular structures at successive time steps (see equation 1).

$$RMSD\left(t_1, t_2\right) = \sqrt{\frac{1}{M}\sum_{i=1}^{N}\left\|r_i\left(t_1\right) - r_i\left(t_2\right)\right\|^2} \quad \text{where } M = \sum_{i=1}^{N} m_i \ . \tag{1}$$

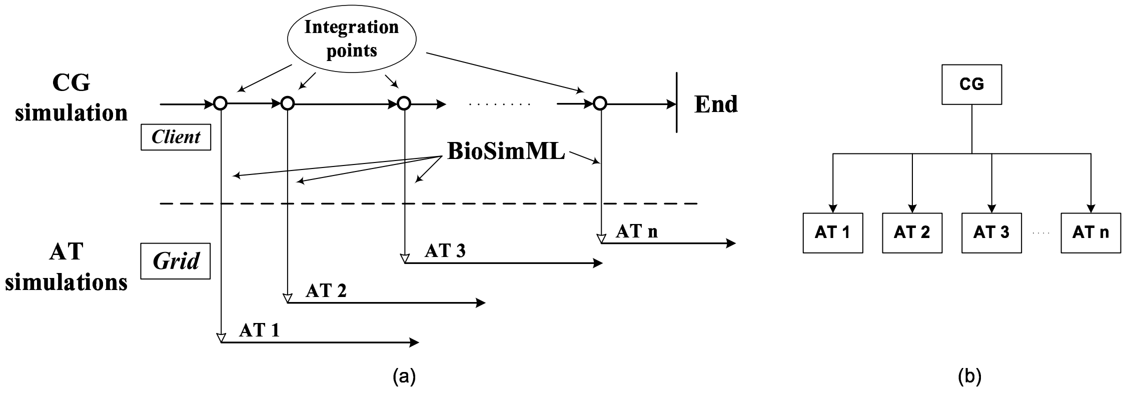


Fig. 3. A multiscale MD simulation model. (a) the workflow of a multiscale simulation with a CG (coarse-grained) simulation running on a client and n AT (atomistic) simulations running on a Grid. (b) the simulation framework. The root is a CG simulation which spawns n AT simulations. Each node contains the components as shown in Fig. 2.

In equation 1, N is the total number of particles. m_i is the mass of particle i and $r_i(t)$ is the position (Cartesian coordinates) of particle i at time t. An integration point is reached when the RMSD has reduced below a predefined value. At this point, the coarse-grained simulation spawns an atomistic (AT) simulation by converting the current coarse-grained molecular structure into an atomistic structure and transmitting it to a computing system where the AT simulation will run. From this point, the coarse-grained simulation and the newly-generated atomistic simulation can both be running concurrently. The coarse-grained simulation can spawn new atomistic simulations at subsequent integration points. As Fig. 3(a) shows, the CG simulation runs on a client machine and spawns an AT simulation at each integration point. Multiple AT simulations can be run in parallel on a Grid. Eventually, the results of these AT simulations can be analysed to determine the best result.

The multiscale simulation model represents a tightly-coupled integration mode. Fig. 3(b) shows the structure of the simulation framework created for this simulation. The CG simulation as the root spawns an AT simulation as offspring through the integration workflow as described in Section III. As the CG simulation can generate multiple AT simulations, a tree-structured framework is built in which each node contains the same components as the Part A and Part B in Fig. 2.

The multiscale simulation model has been implemented on a distributed system which includes a Linux desktop and the NGS. The desktop is used as a client machine to run the CG simulation by GROMACS serial run. AT simulations run on NGS using the computing clusters HPCx (www.hpcx.ac.uk) and NGS-2 (www.ngs.ac.uk/sites/ral/ngs2.html), each AT simulation running in the GROMACS parallel mode. The interactions between the CG and AT simulations are based on the Globus Toolkit 2 (GT2) as NGS is currently using GT2. The simulation framework is deployed on these computing sites under our user directories to create a unified simulation infrastructure across these systems. The data communication between the CG and AT simulations is captured in BioSimML. The BioSimML code shown in Section IV describes an AT simulation. It will be sent from the CG side to an AT side to trigger an AT simulation.

During the integration, the coarse-grained to atomistic data transformation is collaboratively fulfilled by both the CG simulation side and the AT simulation side. The molecular system in this simulation is composed of a protein, 256 DPPC lipids, and around 4000 water molecules. The conversion of the protein requires special software tools, MODELLER [21] and PROCHECK [22], for protein modelling and quality check. We have installed these tools on the client but not provided by NGS. The conversion of all lipids is a time-consuming process, however, we have developed an MPI program to convert all lipids in parallel. Consequently, the conversion of the protein runs on the client machine where the CG simulation has been running, whereas all lipids are converted in parallel on a Grid where an AT simulation is to be run. These two conversions are simultaneously undertaken and thereafter assembled into one molecular system on the AT side. The collaborative data transformation shows an instance of the resource-dependent job distribution supported by the framework. Atomistic water molecules are directly added into the molecular system using the GROMACS solvent model rather than being converted from the coarse-grained structure.

We have run the multiscale simulation model with one CG simulation which spawns three AT simulations. Table I shows the execution times of these simulations and the speedups achieved. In the Table, the CG simulation spawned three AT simulations at timescale points 14ns (nanosecond), 24.8ns, and 36.4ns. Then, each AT simulation ran over a 1.5ns timescale in parallel mode on 32 processors. The CG simulation generated the first AT simulation after running for 1.89 hours that simulated a 14ns timescale. The collaborative conversion of the CG molecular structure to the AT structure took 0.35 hour. The execution time of the AT simulation over a 1.5ns timescale was 5.89 hours. Thus, the total execution time of the CG simulation and the first AT simulation was 8.13 hours which simulated an overall 15.5ns timescale. As a comparison, the Table also shows the execution time of an equivalent pure parallel AT simulation on 32 processors which was 60.86

TABLE I

TIMES AND SPEEDUPS OF THE MULTISCALE SIMULATION MODEL IN COMPARISON WITH PURE PARALLEL ATOMISTIC SIMULATIONS (SPEEDUP=PURE ATOMISTIC RUN TIME / MULTISCALE TOTAL RUN TIME)

Multiscale Simulation	**CG + AT1**		**CG + AT2**		**CG + AT3**	
	Timescale (ns)	**Run time (hours)**	**Timescale (ns)**	**Run time (hours)**	**Timescale (ns)**	**Run time (hours)**
CG	14	1.89	24.8	3.35	36.4	4.37
AT	1.5	5.89	1.5	5.94	1.5	6.04
Data conversion	-	0.35	-	0.37	-	0.34
Total time	15.5	8.13	26.3	9.66	37.9	10.75
Pure atomistic (parallel)	15.5	60.86	26.3	104.15	37.9	152.50
Speedup	7.49		10.78		14.19	

hours for the 15.5ns timescale. Compared to the pure parallel AT simulation, the multiscale simulation achieved a 7.5-fold speedup for the simulation of the 15.5ns timescale. The times of all AT simulations were obtained on HPCx because the best performance of GROMACS parallel run occurs on HPCx using 32 processors in our test as shown in Fig. 4. The Figure shows the simulation speed of GROMACS on HPCx and NGS-2 measured in ns per day which represents the timescale a simulation can achieve in 24 hours. The limited scalability of GROMACS is due to the communication/synchronization pattern and imbalanced workload in its parallel algorithm [23]. This fact confirms the necessity of integrating CG and AT simulations to improve the simulation performance based on GROMACS.

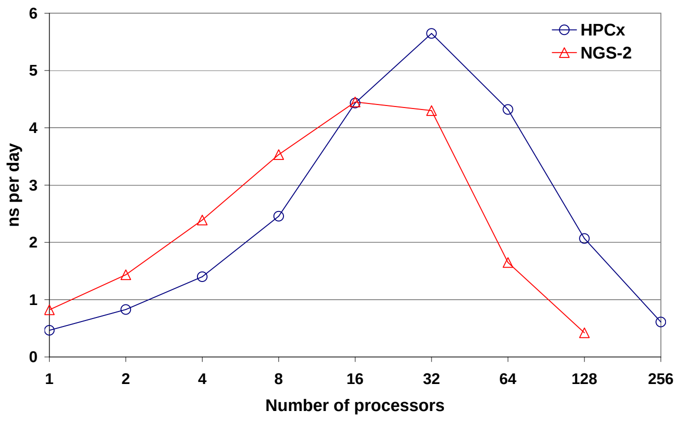


Fig. 4. Simulation speeds of an atomistic simulation using GROMACS parallel run on HPCx and NGS-2

For the accuracy of the multiscale simulation model, the research in [20], [24] has confirmed that the accuracy of a CG simulation is comparable to an equivalent AT simulation. The RMSD and RMSF (root mean square fluctuation) between the molecular structures produced by the equivalent CG and AT simulations are within 5% of each other. As a combination of CG and AT simulations, the multiscale simulation model can obviously also present a comparable accuracy to a pure AT simulation.

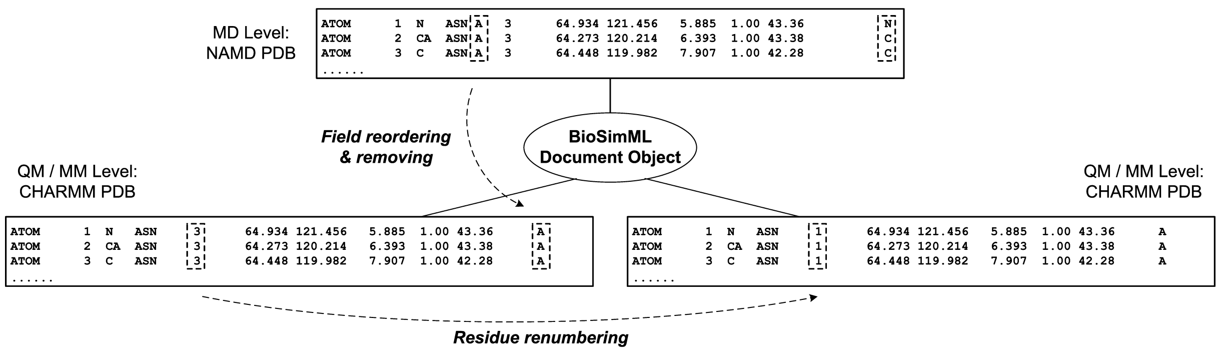
B. Data Transformation between MD and QM/MM Levels

The BioSimML markup language discussed in Section IV provides an XML-based, unified data representation for the integration of multi-level biomolecular simulations. It has been used to assist the transformation of the simulation data between MD and QM/MM levels.

QM/MM simulation is at the most fundamental level in our simulations that models protein reactions. It uses the data of protein conformations produced by MD simulations to select the initial conditions of reaction simulations. In this case, the data of the MD simulation is produced by NAMD. However, the QM/MM simulation needs to use another simulation tool CHARMM. Although both tools accept the protein conformation data in the PDB format, they have some different expectations in how the data is formatted. As Fig. 5 shows, the PDB file generated by NAMD contains a list of ‘A’s at column 22 (the chain ID field, indicated in dashed box) and a list of element symbols (‘N’ and ‘C’) at column 78 (in dashed box). Using this PDB file as input to CHARMM, the chain ID field needs to be moved to column 73 (the segment ID field, see dashed box) and remove column 78. This data transformation can be readily done using BioSimML. Given the PDB file produced by NAMD, the BioSimML handler can parse the PDB file into a BioSimML document object as an intermediate data representation. The textual content of this object is analogous to the `<pdb>` element shown in the BioSimML code in Section IV. Then, the BioSimML handler is applied again to rewrite the elements of the document object to a PDB file in the format required by CHARMM as shown in the *field reordering and removing* step in Fig. 5.

In addition, the QM/MM simulation also needs to normalise the residue sequence numbers with a starting number of ‘1’ in the PDB file. The *residue renumbering* step in Fig. 5 renumbers all residues from the starting number ‘1’ instead of the original number ‘3’ (shown in dashed boxes). This renumbering step is also accomplished via the BioSimML document object by means of the BioSimML handler.

This application shows that BioSimML has provided a standard representation for the widely used data format in molecular simulations. It can facilitate the data transformation between multi-level simulations along with the BioSimML handler and associated utility programs developed in our



```
ATOM      1  N   ASN A  3      64.934 121.456   5.885  1.00 43.36          N
ATOM      2  CA  ASN A  3      64.273 120.214   6.393  1.00 43.38          C
ATOM      3  C   ASN A  3      64.448 119.982   7.907  1.00 42.28          C
......
```

```
ATOM      1  N   ASN    3      64.934 121.456   5.885  1.00 43.36      A
ATOM      2  CA  ASN    3      64.273 120.214   6.393  1.00 43.38      A
ATOM      3  C   ASN    3      64.448 119.982   7.907  1.00 42.28      A
......
```

```
ATOM      1  N   ASN    1      64.934 121.456   5.885  1.00 43.36      A
ATOM      2  CA  ASN    1      64.273 120.214   6.393  1.00 43.38      A
ATOM      3  C   ASN    1      64.448 119.982   7.907  1.00 42.28      A
......
```

Fig. 5. The PDB format transformation between MD and QM/MM levels

research. This application is also a representative of a loosely-coupled integration mode in multi-level simulations.

VI. Conclusions

A multi-level simulation framework has been presented to support the seamless integration of molecular simulations on distinct levels. In coordination with the BioSimML markup language, the framework provides a unified computing infrastructure to implement dynamic coupling and formatted data exchange between different simulation levels. The framework can create an e-Science infrastructure for undertaking biomolecular simulations on Grids. Based on our framework, the experimental simulations have demonstrated high performance gain and efficient data transformation.

We will apply the framework to developing a realistic scenario of integrated simulations spanning across multiple levels using different simulation tools to produce useful results for biological and biochemical research. BioSimML will be extended to provide the constructs for the data representations of other simulation levels and tools.

Acknowledgment

This research is funded by the BBSRC project IntBioSim. The authors thank all collaborators including Graham Riley and Rupert Ford for the proposed hybrid MD simulation model, and Adrian Mulholland and Christo Christov for the QM/MM simulation and data. This work made use of the UK Grid resources of NGS and HPCx.

References

[1] M. Karplus and J. McCammon, "Molecular dynamics simulations of biomolecules," *Nature Structural Biology*, vol. 9, no. 9, pp. 646–652, 2002.

[2] B. Roux and K. Schulten, "Computational studies of membrane channels," *Structure*, vol. 12, no. 9, pp. 1343–1351, 2004.

[3] A. Leach, *Molecular Modelling: Principles and Applications*, 2nd ed. Prentice Hall, 2001.

[4] E. Lindahl, B. Hess, and D. Spoel, "GROMACS 3.0: a package for molecular simulation and trajectory analysis," *Journal of Molecular Modeling*, vol. 7, no. 8, pp. 306–317, 2001.

[5] J. Phillips, R. Braun, W. Wang, J. Gumbart, and et al, "Scalable molecular dynamics with NAMD," *Journal of Computational Chemistry*, vol. 26, no. 16, pp. 1781–1802, 2005.

[6] CHARMM. [Online]. Available: http://www.charmm.org

[7] D. J. Gavaghan, A. C. Simpson, S. Lloyd, D. F. M. Randal, and D. R. S. Boyd, "Towards a grid infrastructure to support integrative approaches to biological research," *Philosophical Transactions of the Royal Society A*, vol. 363, pp. 1829–1841, 2005.

[8] P. Coveney and P. Fowler, "Modelling biological complexity: a physical scientist's perspective," *Journal of Royal Society Interface*, vol. 2, no. 4, pp. 267–280, 2005.

[9] R. Delgado-Buscalioni, P. Coveney, G. Riley, and R. Ford, "Hybrid molecular-continuum fluid models: implementation within a general coupling framework," *Philosophical Transactions of the Royal Society A*, vol. 363, no. 1833, pp. 1975–1985, 2005.

[10] M. Ng, S. Johnston, B. Wu, S. Murdock, K. Tai, H. Fangohr, S. J. Cox, J. W. Essex, M. S. P. Sansom, and P. Jeffreys, "BioSimGrid: Grid-enabled biomolecular simulation data storage and analysis," *Future Generation Computer Systems*, vol. 22, pp. 657–664, 2006.

[11] C. Lloyd, M. Halstead, and P. Nielsen, "CellML: its future, present and past," *Progress in Biophysics & Molecular Biology*, vol. 85, no. 2-3, pp. 433–450, 2004.

[12] Mathematical markup language (MathML) version 2.0. [Online]. Available: http://www.w3.org/TR/MathML2

[13] M. Hucka, A. Finney, H. Sauro, H. Bolouri, J. C. Doyle, and et al, "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models," *Bioinformatics*, vol. 19, no. 4, pp. 524–531, 2003.

[14] Protein Data Bank. [Online]. Available: http://www.rcsb.org

[15] J. Westbrook, N. Ito, H. Nakamura, and et al, "PDBML: the representation of archival macromolecular structure data in XML," *Bioinformatics*, vol. 21, no. 7, pp. 988–992, 2005.

[16] P. E. Bourne, H. M. Berman, B. McMahon, K. D. Watenpaugh, J. Westbrook, and P. M. D. Fitzgerald, "The macromolecular crystallographic information file (mmCIF)," *Methods in Enzymology*, vol. 277, pp. 571–590, 1997.

[17] TINKER. [Online]. Available: http://dasher.wustl.edu/tinker/

[18] RELAX NG. [Online]. Available: http://www.relaxng.org

[19] P. Fowler, K. Balali-Mood, S. Deol, P. Coveney, and M. S. P. Sansom, "Monotopic enzymes and lipid bilayers: a comparative study," *Biochemistry*, vol. 46, no. 11, pp. 3108–3115, 2007.

[20] P. Bond, J. Holyoake, A. Ivetac, S. Khalid, and M. S. P. Sansom, "Coarse-grained molecular dynamics simulations of membrane proteins and peptides," *Journal of Structural Biology*, vol. 157, no. 3, pp. 593–605, 2006.

[21] M. Marti-Renom, A. Stuart, A. Fiser, R. Sánchez, and et al, "Comparative protein structure modeling of genes and genomes," *Annual Review of Biophysics and Biomolecular Structure*, vol. 29, pp. 291–325, 2000.

[22] R. Laskowski, M. MacArthur, D. Moss, and J. Thornton, "PROCHECK: a program to check the stereochemical quality of protein structures," *Journal of Applied Crystallography*, vol. 26, no. 2, pp. 283–291, 1993.

[23] Z. Cui and Z. Lan, "Performance study of molecular dynamics application GROMACS," in *Proc. Parallel and Distributed Computing and Systems (PDCS 2004)*, MIT Cambridge, USA, Nov. 2004.

[24] P. Bond and M. S. P. Sansom, "Insertion and assembly of membrane proteins via simulation," *Journal of the American Chemical Society*, vol. 128, no. 8, pp. 2697–2704, 2006.