BIOMOLECULAR SIMULATION DATA MANAGEMENT IN HETEROGENEOUS ENVIRONMENTS

Julien Charles Victor Thibault

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Biomedical Informatics

The University of Utah

December 2014

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Julien Charles Victor Thibault**

has been approved by the following supervisory committee members:

Julio Cesar Facelli	, Chair	**4/2/2014** Date Approved
Thomas E. Cheatham	, Member	**3/31/2014** Date Approved
Karen Eilbeck	, Member	**4/3/2014** Date Approved
Lewis J. Frey	, Member	**4/2/2014** Date Approved
Scott P. Narus	, Member	**4/4/2014** Date Approved

And by **Wendy W. Chapman**, Chair of

the Department of **Biomedical Informatics**

and by David B. Kieda, Dean of The Graduate School.

ABSTRACT

Over 40 years ago, the first computer simulation of a protein was reported: the atomic motions of a 58 amino acid protein were simulated for few picoseconds. With today's supercomputers, simulations of large biomolecular systems with hundreds of thousands of atoms can reach biologically significant timescales. Through dynamics information biomolecular simulations can provide new insights into molecular structure and function to support the development of new drugs or therapies. While the recent advances in high-performance computing hardware and computational methods have enabled scientists to run longer simulations, they also created new challenges for data management. Investigators need to use local and national resources to run these simulations and store their output, which can reach terabytes of data on disk. Because of the wide variety of computational methods and software packages available to the community, no standard data representation has been established to describe the computational protocol and the output of these simulations, preventing data sharing and collaboration. Data exchange is also limited due to the lack of repositories and tools to summarize, index, and search biomolecular simulation datasets.

In this dissertation a common data model for biomolecular simulations is proposed to guide the design of future databases and APIs. The data model was then extended to a controlled vocabulary that can be used in the context of the semantic web. Two different approaches to data management are also proposed. The iBIOMES repository offers a distributed environment where input and output files are indexed via common data elements. The repository includes a dynamic web interface to summarize, visualize, search, and download published data. A simpler tool, iBIOMES Lite, was developed to generate summaries of datasets hosted at remote sites where user privileges and/or IT resources might be limited. These two informatics-based approaches to data management offer new means for the community to keep track of distributed and heterogeneous biomolecular simulation data and create collaborative networks.

To My Family

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AIMD – ab initio molecular dynamics

API – application programming interface

AVU – attribute-value-unit

CDE – common data element

CML – Chemistry Markup Language

CPU – central processing unit

CSD – Cambridge Structural Database

JSP – JavaServer Pages

GPU – graphics processing unit

HTML – HyperText Markup Language

MD – molecular dynamics

OWL – Web Ontology Language

PDB – Protein Data Bank

QM – quantum mechanics

QM/MM – quantum mechanics/molecular mechanics

SEBOMD – semi-empirical Born-Oppenheimer molecular dynamics

SEMD – semi-empirical molecular dynamics

SKOS – Simple Knowledge Organization System

XML – eXtensible Markup Language

XSL – eXtensible Stylesheet Language

ACKNOWLEDGMENTS

This work would have not been possible without the help from many people at the University of Utah and the support from my family back in France.

In June 2010 I joined Thomas Cheatham's lab in the Medicinal Chemistry department. He was not only one of my committee members, but also a mentor for over 4 years. I thank him for giving me the opportunity to work on several informatics-based projects outside his usual line of research, and for funding me during that time. I also want to thank the whole lab for giving me insights into the field of biomolecular simulations and for helping me understand researchers' needs: thanks to Rodrigo Galindo, Niel Henriksen, Kiumars Shahrokh, Daniel Roe, Christina Bergonzo, and Sean Cornillie.

I want to thank my committee chair, Julio Facelli, who kept me on track with my dissertation and made sure I always could see the big picture in my research. He also provided me with invaluable resources through the Center for High-Performance Computing at the University of Utah, including computer time allocations and hardware and software support. Thanks to the CHPC staff for all the help and support, especially Ron Price, Anita Orendt, Wim Cardoen, and Brian Haymore who were involved in the implementation of some of the projects presented here.

I would like to thank my other committee members, Lewis Frey, Karen Eilbeck, and Scott Narus who provided me with valuable feedback to conduct my research. I also would like to thank Stephane Meystre for giving me the opportunity to work on several projects involving clinical Natural Language Processing (NLP) and for providing me with funding during my first year as a Ph.D. student. I would also like to thank the faculty, students, and staff at the Department of Biomedical Informatics who have provided a great environment for learning and research.

Finally I would like to thank my parents and my sister, who supported me during all these years in school. Although they have been physically distant, they have always found a way to encourage me when it was needed.

CHAPTER 1

INTRODUCTION

Problem statement

Biomolecular simulations aim to simulate large biomolecular systems in silico to provide insight into biological structure and function through molecular dynamics. They can be used for prediction purposes as a screening step for experiments, or to complement experimental studies by providing transition information between representative structural conformations. With recent advances in computational hardware[1-3] and algorithmic techniques,[4] simulations can now reach time scales that are biologically significant to study dynamic processes such as protein folding. The data generated by these simulations are overwhelming because of the storage requirements and the heterogeneity of the computational methods being used. The data are highly unorganized: each computational experiment can consist of hundreds of input (e.g., system topology, simulation parameters) and output (e.g., atom trajectories, energies, temperatures) files in different formats, and following user-specific naming conventions. The data also tend to be scattered among distributed resources, at the researcher's home institution and at national computing centers where the data are generated. It becomes nontrivial even for primary investigators to keep track of their data, especially when the data were generated by past students or collaborators, who might use different methods, software packages, and file naming conventions. New tools are needed by researchers to catalog these files and provide a structured view of the data to enable data browsing, searching and mining at the level of the lab, and to enable data exchange with collaborators or the larger community.

Main objectives

This dissertation focuses on the development of computable data models for biomolecular simulations and management tools to summarize, track, and share datasets stored in heterogeneous and distributed environments. The specific aims pursued in this research are presented in the next paragraphs.

Aim 1

Hypothesis: A common data model can represent the computational protocols used in biomolecular simulations.

Research question 1.1: Can a common model represent the variety of methods used in biomolecular simulations (i.e., ab initio, semi-empirical, and empirical methods)?

Research question 1.2: Can such a model be used to develop new databases and/or Application Programming Interfaces (API)?

Research question 1.3: Can such a model be used to develop a controlled vocabulary that can be used in a semantic web context?

For this aim a data model and set of dictionaries were designed to address the representation of computational models (e.g., molecular dynamics, quantum mechanics), parameters, authorship, molecular systems (biomolecules and chemical compounds), computing environments, and files (input and output). The data model was used in different prototypes to show its applicability to databases and APIs for biomolecular simulation data management. The data model and dictionaries were then used to create a controlled vocabulary, in the form of a database similar to the UMLS metathesaurus,[5] which was extended to a Simple Knowledge Organization System[6] (SKOS) and an OWL ontology.

Aim 2

Hypothesis: A repository can be built to store, index, and present biomolecular simulation data distributed among multiple resources.

Research question 2.1: Can current technology be used to develop a distributed repository for biomolecular simulation input and output files?

Research question 2.2: Can the repository support data queries using common data elements?

A repository (iBIOMES) was designed and implemented to integrate a distribute file system where files are indexed using common data elements. It includes a dynamic web interface to summarize, visualize, search, and download published data.

Aim 3

Hypothesis: A simple tool can be developed to track and share biomolecular simulation data hosted in heterogeneous environments where user privileges and IT support are limited.

Research question 3.1: Can a single tool summarize heterogeneous biomolecular simulation datasets using a common data model?

Research question 3.2: Can this tool be deployed and used in limited settings where user privileges and IT support are limited?

A simple tool (iBIOMES Lite) was created to generate XML and HTML summaries of biomolecular simulation datasets. A set of file parsers is used to automatically create a representation of the computational protocol based on a common data model.

Dissertation outline

Chapter 2 provides background information about biomolecular simulations, data challenges, and current environments available to manage and share these data. The next four chapters address the three research aims introduced earlier. Chapter 3 and 4 provide the basis for a common representation of biomolecular simulation data. Chapter 3 focuses on the design of a logical data model and a set of dictionaries for database and API design while Chapter 4 focuses on the development of a controlled vocabulary that can be used in a semantic web context. Chapter 5 introduces iBIOMES, a distributed repository architecture for simulation data publication. Chapter 6 introduces iBIOMES Lite, a light-weight tool that can be deployed in limited settings to summarize and share simulation protocols and results. Finally in Chapter 7 the results of the research are summarized and discussed.

References

1. Narumi, T.; Ohno, Y.; Okimoto, N.; Suenaga, A.; Yanai, R.; Taiji, M. A High-Speed Special-Purpose Computer for Molecular Dynamics Simulations: MDGRAPE-3. In NIC Workshop, 2006; 2006; Vol. 34; pp 29-36.

2. Shaw, D. E.; Dror, R. O.; Salmon, J. K.; Grossman, J. P.; Mackenzie, K. M.; Bank, J. A.; Young, C.; Deneroff, M. M.; Batson, B.; Bowers, K. J.; Chow, E.; Eastwood, M. P.; Ierardi, D. J.; Klepeis, J. L.; Kuskin, J. S.; Larson, R. H.; Lindorff-Larsen, K.; Maragakis, P.; Moraes, M. A.; Piana, S.; Shan, Y.; Towles, B., In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*; ACM: Portland, Oregon, 2009, pp 1-11.

3. Le Grand, S.; Götz, A. W.; Walker, R. C., SPFP: Speed Without Compromise—A Mixed Precision Model for GPU Accelerated Molecular Dynamics Simulations. *Comput. Phys. Commun.* **2013**, 184, 374-380.

4. Schlick, T., Molecular Dynamics-Based Approaches for Enhanced Sampling of Long-Time, Large-Scale Conformational Changes in Biomolecules. *F1000 biology reports* **2009**, 1, 51.

5. Bodenreider, O., The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic Acids Res.* **2004**, 32, D267.

6. Simple Knowledge Organization System (SKOS) reference. http://www.w3.org/TR/skos-reference/

CHAPTER 2

BACKGROUND

Biomolecular simulations

Introduction

Biomolecular simulations aim to simulate the motions of complex biomolecular systems characterized at the atomic level. Simulated systems include proteins,[1] nucleic acids (DNA, RNA),[2, 3] lipids,[4, 5] and carbohydrates.[6, 7] Through dynamics, simulations can provide new insights into molecular structure and function.[8] They can be used to supplement existing experiments, guide the design of new experiments, or provide insights that might not be determined experimentally because of current protocol limitations. Another major application of biomolecular simulations is the study of interactions between biomolecules and ligands in the context of drug discovery.[9] Understanding binding affinities between receptor and ligand is a critical component to develop better drugs, therapies, catalysts and nanotechnology.[8, 10, 11] Simulation implementations have evolved along with advances in hardware and software technology and it is now possible to use more complex and accurate models to study the dynamics of biomolecules.[12, 13] As the implementations of biomolecular simulations software evolve, developers need to keep validating their models and their specific implementations using experimental data[14-17] (e.g., crystal or NMR structures) or alternative computational methods that provide highly accurate results.[18-20] Validation of simulation output is a necessary step for users as well.[16, 21] The large amount of data generated by these simulations must be checked for errors, analyzed, and interpreted to draw conclusions that have a biological meaning.

Molecular dynamics

Molecular dynamics (MD) is arguably the most popular class of methods for biomolecular simulations today. MD methods use Newton's equations of motion to compute the atomic positions over discrete time steps, called trajectories. The simulated molecule or set of molecules is represented by a system of interacting particles. For each particle *i* in a system constituted by *N* particles, Newton's equations of motion define the force $\vec{F}_i$ acting on the particle as

$$\vec{F}_i = m_i \vec{a}_i = -\vec{\nabla}_i U \tag{1}$$

where m_i is the mass of the particle *i*, $\vec{a}_i$ its acceleration, and $-\vec{\nabla}_i U$ the gradient of the potential energy.

The acceleration can then be expressed as

$$\vec{a}_i = \frac{d^2\vec{r}_i}{dt^2} = -\frac{1}{m_i}\frac{d\vec{U}}{dr_i} \tag{2}$$

where r_i is the position of the particle.

Using a Taylor series expansion, the position of the particle along a single dimension *x* after an increment in time Δt can be described as

$$x(t + \Delta t) = x(t) + \frac{dx(t)}{dt}\Delta t + \frac{d^2x(t)}{dt^2}\Delta t^2 + \cdots \tag{3}$$

Several integration algorithms use a truncated version of this series to integrate the equations of motions using small steps in time (Δt). For example, in the simple Verlet algorithm[22] the position x of a particle at the instant (t+Δt) is given by:

$$x(t+\Delta t) = 2x(t) - x(t-\Delta t) + \frac{d^2x(t)}{dt^2}\Delta t^2 \tag{4}$$

Therefore, using information about the previous time steps and Equation (2) one can determine the position of the particle i at the instant t if the potential energy is known.

In classical MD, the potential energy is derived from molecular mechanics (MM) principles. Each particle in the system is represented by a sphere with a certain radius (van der Waals radius), polarizability, and charge, while bonds are represented by springs. The potential energy is mathematically described through a force field, a mathematical function that is parameterized to enable different set of parameters for different types of particles. Force fields describe both bonded interactions (between atoms linked by covalent bonds) and nonbonded interactions (long-range interactions). Bonded interactions can be described through terms that represent bonds and angles between the different particles for example. The nonbonded interactions typically include van der Waals forces and electrostatic interactions.

In all-atom MD, each atom is represented by a single particle in the system. The force field parameter set is then dependent on the atom type, which is typically defined by the corresponding atomic element (e.g., Carbon, Oxygen), but also by the electronic configuration of the atom. In coarse-grain (CG) MD[23] each particle in the system represents a group of atoms rather than an individual atom. For example, each residue (e.g., amino acid in protein) can be approximated as a bead, dramatically reducing the cost of calculations compared to an all-atom representation. Unfortunately CG representations can also lead to more simulation inaccuracies. For example, the side chain motions cannot be well described, although they are known to have an influence on polymers' properties. Independently from the granularity of the representation, most force field parameter sets tend to be domain specific. For example, a given force field parameter set might be adapted to protein modeling[24-26] while another one might be recommended for nucleic acid simulations.[19, 27]

Quantum chemistry

One of the limitations of classical MD is that chemical reactions where bonds form and break cannot be represented. To overcome these limitations the potential energy can be calculated using a quantum mechanics (QM) method to provide an electronic description of the system. In quantum chemistry the electronic structure of the atoms is explicitly described through Schrödinger's equations. The spatial distribution and energy of an electron can be defined though molecular orbitals, which can be described through a set of wave functions: the basis sets. Different levels of theory are available to approximate the selected basis set and find a discrete set of solutions to the Schrödinger equation. Popular methods include Hartree-Fock (HF) and post-Hartree-Fock methods (e.g., Configuration Interaction, Moller-Plesset, Coupled-Cluster), multi-reference methods, and Density Functional Theory (DFT).

In ab initio molecular dynamics[28] (AIMD) these methods are used to replace the MM force field and compute the potential energy of the system using a quantum approach. Because of the computational cost of quantum methods, AIMD methods are only used on small biomolecular systems and reduced time scales compared to classical

MD. In semi-empirical MD[29] (SEMD) the quantum methods that are used make many approximations using empirical formulae.[30] These approaches provide a less accurate electronic description of the system but they can greatly reduce the cost of the QM calculations.

The role of quantum chemistry in biomolecular simulations is not limited to quantum MD applications. Many MM force field parameters set developments for example are guided by quantum calculations, which can give very accurate results on small test cases and help fitting parameters.[18-20] Quantum chemistry can also be used in hybrid QM/MM approaches where the system is partitioned into a QM region and an MM region.[31] Assuming that the QM region is fairly small and targets a region of interest (e.g., binding site of a protein) QM/MM simulations can combine the speed of classical MD and the level of accuracy of QM methods.

Computing environment

Software packages

A wide variety of MD and QM parallel codes are available to the scientific community. AMBER,[32] CHARMM,[33] NAMD,[34] GROMACS,[35] Desmond,[36] and GROMOS[37] are some of the most popular MD simulation codes in use today to simulate proteins, nucleic acids, or even larger molecules. Gaussian,[38] NWChem,[39], Q-Chem,[40] GAMESS,[41] Jaguar,[42] or VASP[43] on the other hand, are popular QM packages, typically used to study small molecules such as drug compounds. Some of these software packages also offer QM/MM capabilities, either by implementing both MD and QM engines, or by allowing external engines to interface with their code.

Many tools are available to analyze the output of the simulations, compare the results to experimental data, and possibly generate new hypotheses. Visualization tools such as VMD[44] or Chimera[45] feature 3D rendering of molecules and visualization of MD trajectories through animations. A more quantitative analysis of the output can be performed with programs like CPPTRAJ[46] or MD-specific scripting libraries (e.g., VMD's Tcl capability, MDAnalysis[47]) to pinpoint anomalies, evaluate differences with other simulations or experimental datasets, and identify events of potential interest.

Because of the wide variety of software packages and computational methods available to the community, no standard format has been adopted to store or describe simulation results. Various cheminformatics projects have emerged, aiming to facilitate computational chemistry data exchange. The Blue Obelisk effort, for example, aims to provide informatics tools with the concepts of Open Data, Open Standards and Open Source in mind, to facilitate collaboration between chemists.[48] Projects such as the Chemistry Markup Language (CML[49, 50]) and the OpenBabel[51] data converter are part of this effort to distribute free tools to the community to encourage the usage of standard data formats. For now these tools are mostly limited to the representation of experimental and quantum chemistry, and only few legacy software packages are adopting them.[52]

High-performance computing hardware

All these packages keep evolving as new hardware allows the implementation of more complex algorithms and numerical techniques. The simulation engines provided by these software packages are very demanding computationally and cannot be run on regular desktop computers. CPU clusters composed of hundreds of computational nodes are today's common computing platform for biomolecular simulations. Despite their computational power, simulations usually have to run for weeks or months to reach time scales that are biologically significant. More modern high-performance computing (HPC) hardware, such as general-purpose Graphics Processor Units (GPUs), is now used in conjunction with CPU nodes to accelerate the computations.[13, 53] Specialized hardware, such as MDGRAPE[54, 55] or Anton,[56] is specifically designed to run molecular dynamics simulations. These machines are usually much faster than general-purpose HPC hardware, but their usage is also limited to the simulation model their architecture supports and they are usually not widely available to researchers.

Data storage

While advances in hardware have allowed simulations of larger systems using longer time scales, they also created a tsunami of data researchers have to store and analyze. Today's simulation output can easily reach terabytes (TB) of data on disk. Most of these data represent the MD trajectories: the time series of the 3D coordinates of each atom in the system. Even though the output can be compressed[57, 58] or stripped from unnecessary information (e.g., remove solvent molecules from the system), data storage and transfer (between national computing centers and home institutions for example) remains a bottleneck. Simulation archiving becomes a necessity if researchers want to keep track of model evolutions and simulation output changes. It also becomes necessary for researchers to adopt new approaches to expose their existing datasets to build collaborative networks and share data with the community. The number of tools for biomolecular simulation data management is currently limited because of the amount of data that need to be stored and described, and because of the heterogeneity of the data due to the wide variety of software packages and computational methods available. In the next sections we present previous projects that aim to develop standard data formats and infrastructures for structural and dynamics data exchange in the experimental and the computational communities.

Data sharing

Experimental data

One of the largest open sources for experimental structures is the Protein Data Bank (PDB),[59] hosted by the Research Collaboratory for Structural Bioinformatics (RCSB). The PDB is widely used by the biomolecular simulation community to validate computational results and to create the initial structures for dynamics runs. While PDB is one of the main resources for experimental structures, no information about dynamics (e.g., MD trajectories) is available, and search capabilities are limited (molecule name, author, ligand, and sequence). Other structural databases, such as the Cambridge Structural Database[60] (CSD), provide more search capabilities, but at a certain financial cost. Several open databases for small chemical molecules exist as well. PubChem provides access to millions of compounds, substances, and bioassays.[61] The database can be searched using advanced queries based on chemical structure, names, and properties (e.g., hydrogen bond donor and acceptor count). ChEMBL is a database of drug-like bioactive compounds.[62] Assays from different sources are represented through a common data model to enable computerized data mining and drug discovery. The ChEMBL database was recently integrated into the semantic web[63] to facilitate inferences with external web resources such as ChemSpider.[64]

Molecular simulations

The BioSimGrid project[65, 66] tackles the simulation data storage problem through a specialized grid. This infrastructure offers secured data deposition and retrieval services. BioSimGrid is supported by a Grid-based architecture to connect distributed relational databases that stores not only biomolecular simulation metadata (e.g., author, software, method) but also molecule topology and trajectory information. Simulation data can be deposited, retrieved, and processed though a Python script environment and a web interface. A prototype of BioSimGrid was deployed in the UK to connect multiple e-Science centers but the current status of the project is uncertain. The code is now available at http://sourceforge.net/projects/biosimgrid/. The Dynameomics project[67] aims to create the largest repository of protein folding simulations. The repository currently indexes about 11,000 simulations of over 2,000 distinct proteins. In order to achieve this, the simulation and analysis workflow had to be computerized, and data warehousing issues had to be addressed. Each atom trajectory is stored in a database, along with metadata about the simulation and the target molecule. Data retrieval was optimized by creating multiple instances of Microsoft SQL at each physical server, and making use of SQL views. The database is also supplemented with a 3D index to speed up nearest neighbor searches[68]. This architecture seems adapted to the authors' particular needs but they note that changes in their database schema could be costly as SQL views would have to be updated and data moved around. A limitation of this project is that the data are not currently open for queries. Access through SQL queries can be requested but one should have prior knowledge about the database schema to obtain the information of interest. A web service interface (SOAP) is in development and might facilitate data integration into external systems. Both BioSimGrid and Dynameomics are limited by the way they store atoms' coordinates. With the advances in high-performance computing, it is now possible to run millisecond simulations, resulting in GB or TB of data. Organizing these data into a relational database is expensive: specialized trajectory compression and indexing techniques are required and new analysis tools need to be developed since most of the current ones are only applicable to file-based trajectories.

Other projects focus on more complex infrastructures that aim to provide a single platform for simulation execution, data storage and postprocessing. The eMinerals project[69] aims to study mineralogical processes through molecular simulations. It is supported by a computational and data minigrid. Data resources are managed by the Storage Resource Broker (SRB),[70] which creates a virtual file repository for the organization. A central metadata catalog (MCAT) stores information about the distributed files and can store associated user-defined metadata. The compute resources and job submissions are managed through Globus and Condor,[71] Several scientific projects[72] showed the benefits of this minigrid implementation. MoDEL[73] (Molecular Dynamics Extended Library) is a large simulation repository, and part of an integrated platform that initially focused on protein simulations. Users set up their simulations via the MDWeb web portal,[74] which automatically takes care of many of the steps necessary to prepare the initial structure (e.g., model downloaded from the PDB) for production MD runs. The simulation jobs are submitted to a supercomputing center and results are centralized into a repository accessible via the web interface for data retrieval and postprocessing. External and local analysis tools such as Ptraj[46] were integrated into the environment to enable trajectory analysis. The public MoDEL database currently indexes 1,700 simulations of proteins and is available at http://mmb.pcb.ub.es/MoDEL/. The MDWeb environment now also provides a computational workflow to study nucleic acids.[75] In these types of integrated environments the simulation runs can be monitored and resulting data can be indexed with accurate metadata. The underlying architectures tend to be very complex and expensive since they require computational resources to run batch jobs, storage resources to manage the resulting datasets, and IT support. At this point one of the main limitations is that external data cannot be published to these platforms. In these integrated environments provenance metadata is generated based on the input provided directly within the environment. Publication of raw data generated outside these environments would require some parsing mechanism[66, 76] to extract the metadata and provide a description of the associated files that fits their data model. Since most researchers currently use resources available at their home institutions or via national computing centers, architectural changes would have to be made to enable the use of these environments as collaborative repositories.

Dissertation

In this dissertation the problem of biomolecular simulation data management is tackled using design criteria informed by previous work published by researchers in the field. First the set of management tools presented here are not tied to the computational component used to run biomolecular simulations, unlike a full workflow-based environment such as MDWeb. This means that the tools are not dependent on the way the simulation data are generated, leaving researchers with the ability to use the computational resources they are already using (e.g., local machine, high-performance computing centers). In order for the tools to be aware of the data, the data need to be "published" to a management system. Publication includes data indexing using provenance metadata and data copy if the management system is not installed where the original data reside (e.g., community-level repository). Users should be able to deploy these tools on heterogeneous platforms – i.e., various types of storage resources that can be distributed over the network – or have the means to access them remotely (e.g., command-line or web interface). A federated approach is used to aggregate distributed resources and enable seamless searches via a single entry point. Java is used to enable deployment and usage of these tools on a variety of operating systems. The management systems presented here are also meant to be context- and method- independent. Using a model-driven approach, the simulation protocol can be used to computationally describe and index data generated by a wide spectrum of methods and software packages, enabling the description of various studies (e.g., quantum calculations on small drug compounds, protein folding simulations). In this work the data model is used to create detailed summaries via the iBIOMES Lite tool and index raw data – i.e., the files – in the context of data exchange and collaboration via the iBIOMES repository.

References

1. Shaw, D. E.; Maragakis, P.; Lindorff-Larsen, K.; Piana, S.; Dror, R. O.; Eastwood, M. P.; Bank, J. A.; Jumper, J. M.; Salmon, J. K.; Shan, Y.; Wriggers, W., Atomic-Level Characterization of the Structural Dynamics of Proteins. *Science* **2010**, 330, 341-6.

2. Lavery, R.; Zakrzewska, K.; Beveridge, D.; Bishop, T. C.; Case, D. A.; Cheatham, T., III; Dixit, S.; Jayaram, B.; Lankas, F.; Laughton, C.; Maddocks, J. H.; Michon, A.; Osman, R.; Orozco, M.; Perez, A.; Singh, T.; Spackova, N.; Sponer, J., A Systematic Molecular Dynamics Study of Nearest-Neighbor Effects on Base Pair and Base Pair Step Conformations and Fluctuations in B-DNA. *Nucleic Acids Res.* **2010**, 38, 299-313.

3. Sponer, J.; Cang, X.; Cheatham, T. E., 3rd, Molecular Dynamics Simulations of G-DNA and Perspectives on the Simulation of Nucleic Acid Structures. *Methods* **2012**, 57, 25-39.

4. Delemotte, L.; Tarek, M., Molecular Dynamics Simulations of Lipid Membrane Electroporation. *J. Membr. Biol.* **2012**, 245, 531-43.

5. Lupyan, D.; Mezei, M.; Logothetis, D. E.; Osman, R., A Molecular Dynamics Investigation of Lipid Bilayer Perturbation by PIP2. *Biophysical journal* **2010**, 98, 240-247.

6. Perić-Hassler, L.; Hansen, H. S.; Baron, R.; Hünenberger, P. H., Conformational Properties of Glucose-Based Disaccharides Investigated Using Molecular Dynamics Simulations with Local Elevation Umbrella Sampling. *Carbohydrate research* **2010**, 345, 1781-1801.

7. Payne, C. M.; Bomble, Y. J.; Taylor, C. B.; McCabe, C.; Himmel, M. E.; Crowley, M. F.; Beckham, G. T., Multiple Functions of Aromatic-Carbohydrate Interactions in a Processive Cellulase Examined with Molecular Simulation. *J. Biol. Chem.* **2011**, 286, 41028-41035.

8. Dror, R. O.; Dirks, R. M.; Grossman, J. P.; Xu, H.; Shaw, D. E., Biomolecular Simulation: a Computational Microscope for Molecular Biology. *Annu. Rev. Biophys.* **2012**, 41, 429-452.

9. Durrant, J. D.; McCammon, J. A., Molecular Dynamics Simulations and Drug Discovery. *BMC biology* **2011**, 9, 71.

10. Alonso, H.; Bliznyuk, A. A.; Gready, J. E., Combining Docking and Molecular Dynamic Simulations in Drug Design. *Med. Res. Rev.* **2006**, 26, 531-568.

11. Klein, M. L.; Shinoda, W., Large-Scale Molecular Dynamics Simulations of Self-Assembling Systems. *Science* **2008**, 321, 798-800.

12. Schlick, T.; Collepardo-Guevara, R.; Halvorsen, L. A.; Jung, S.; Xiao, X., Biomolecular Modeling and Simulation: A Field Coming of Age. *Quarterly reviews of biophysics* **2011**, 44, 191-228.

13. Giupponi, G.; Harvey, M. J.; De Fabritiis, G., The Impact of Accelerator Processors for High-Throughput Molecular Modeling and Simulation. *Drug discovery today* **2008**, 13, 1052-8.

14. Showalter, S. A.; Brüschweiler, R., Validation of Molecular Dynamics Simulations of Biomolecules Using NMR Spin Relaxation as Benchmarks: Application to the AMBER99SB Force Field. *J. Chem. Theory Comput.* **2007**, 3, 961-975.

15. Lindorff-Larsen, K.; Maragakis, P.; Piana, S.; Eastwood, M. P.; Dror, R. O.; Shaw, D. E., Systematic Validation of Protein Force Fields Against Experimental Data. *PloS one* **2012**, 7, e32131.

16. van Gunsteren, W. F.; Mark, A. E., Validation of Molecular Dynamics Simulation. *J. Chem. Phys.* **1998**, 108, 6109-6116.

17. Laskowski, R. A.; MacArthur, M. W.; Thornton, J. M., Validation of Protein Models Derived from Experiment. *Curr. Opin. Struct. Biol.* **1998**, 8, 631-639.

18. Kaminski, G. A.; Friesner, R. A.; Tirado-Rives, J.; Jorgensen, W. L., Evaluation and Reparametrization of the OPLS-AA Force Field for Proteins via Comparison with Accurate Quantum Chemical Calculations on Peptides. *J. Phys. Chem. B* **2001**, 105, 6474-6487.

19. Zgarbová, M.; Otyepka, M.; Šponer, J. i.; Mládek, A. t.; Banáš, P.; Cheatham III, T. E.; Jurecka, P., Refinement of the Cornell et al. Nucleic Acids Force Field Based on Reference Quantum Chemical Calculations of Glycosidic Torsion Profiles. *J. Chem. Theory Comput.* **2011**, 7, 2886-2902.

20. Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A., Development and Testing of a General Amber Force Field. *J. Comput. Chem.* **2004**, 25, 1157-74.

21. Murdock, S. E.; Tai, K.; Ng, M. H.; Johnston, S.; Wu, B.; Fangohr, H.; Laughton, C. A.; Essex, J. W.; Sansom, M. S., Quality Assurance for Biomolecular Simulations. *J. Chem. Theory Comput.* **2006**, 2, 1477-1481.

22. Verlet, L., Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Physical review* **1967**, 159, 98.

23. Takada, S., Coarse-Grained Molecular Simulations of Large Biomolecules. *Curr. Opin. Struct. Biol.* **2012**, 22, 130-7.

24. Lindorff‐Larsen, K.; Piana, S.; Palmo, K.; Maragakis, P.; Klepeis, J. L.; Dror, R. O.; Shaw, D. E., Improved Side‐Chain Torsion Potentials for the Amber ff99SB Protein Force Field. *Proteins: Structure, Function, and Bioinformatics* **2010**, 78, 1950-1958.

25. de Jong, D. H.; Singh, G.; Bennett, W. D.; Arnarez, C.; Wassenaar, T. A.; Schäfer, L. V.; Periole, X.; Tieleman, D. P.; Marrink, S. J., Improved Parameters for the Martini Coarse-Grained Protein Force Field. *J. Chem. Theory Comput.* **2012**, 9, 687-697.

26. Best, R. B.; Zhu, X.; Shim, J.; Lopes, P. E.; Mittal, J.; Feig, M.; MacKerell Jr, A. D., Optimization of the Additive CHARMM All-Atom Protein Force Field Targeting Improved Sampling of the Backbone ϕ, ψ and Side-Chain $\chi1$ and $\chi2$ Dihedral Angles. *J. Chem. Theory Comput.* **2012**, 8, 3257-3273.

27. Baker, C. M.; Anisimov, V. M.; MacKerell, A. D., Jr., Development of CHARMM Polarizable Force Field for Nucleic Acid Bases Based on the Classical Drude Oscillator Model. *J. Phys. Chem. B* **2011**, 115, 580-96.

28. Marx, D.; Hutter, J., Ab Initio Molecular Dynamics: Theory and Implementation. *Modern methods and algorithms of quantum chemistry* **2000**, 1, 301-449.

29. Stewart, J. J.; Davis, L. P.; Burggraf, L. W., Semi ‐ Empirical Calculations of Molecular Trajectories: Method and Applications to Some Simple Molecular Systems. *J. Comput. Chem.* **1987**, 8, 1117-1123.

30. Bredow, T.; Jug, K., Theory and Range of Modern Semiempirical Molecular Orbital Methods. *Theoretical Chemistry Accounts* **2005**, 113, 1-14.

31. Senn, H. M.; Thiel, W., QM/MM Methods for Biomolecular Systems. *Angew. Chem. Int. Ed. Engl.* **2009**, 48, 1198-229.

32. Case, D. A.; Cheatham, T. E., 3rd; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M., Jr.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J., The Amber Biomolecular Simulation Programs. *J. Comput. Chem.* **2005**, 26, 1668-1688.

33. Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D., CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics Calculations. *J. Comput. Chem.* **1983**, 4, 187-217.

34. Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K., Scalable Molecular Dynamics with NAMD. *J. Comput. Chem.* **2005**, 26, 1781-1802.

35. Pronk, S.; Pall, S.; Schulz, R.; Larsson, P.; Bjelkmar, P.; Apostolov, R.; Shirts, M. R.; Smith, J. C.; Kasson, P. M.; van der Spoel, D.; Hess, B.; Lindahl, E., GROMACS 4.5: A High-Throughput and Highly Parallel Open Source Molecular Simulation Toolkit. *Bioinformatics* **2013**, 29, 845-54.

36. Bowers, K. J.; Chow, E.; Xu, H.; Dror, R. O.; Eastwood, M. P.; Gregersen, B. A.; Klepeis, J. L.; Kolossvary, I.; Moraes, M. A.; Sacerdoti, F. D. Scalable Algorithms for Molecular Dynamics Simulations on Commodity Clusters. In SC 2006 Conference, Proceedings of the ACM/IEEE, 2006; IEEE: 2006; pp 43-43.

37. Christen, M.; Hünenberger, P. H.; Bakowies, D.; Baron, R.; Bürgi, R.; Geerke, D. P.; Heinz, T. N.; Kastenholz, M. A.; Kräutler, V.; Oostenbrink, C., The GROMOS Software for Biomolecular Simulation: GROMOS05. *J. Comput. Chem.* **2005**, 26, 1719-1751.

38. Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. *Gaussian 09*, Revision C. 01; Gaussian, Inc: Wallingford, CT, 2009.

39. Valiev, M.; Bylaska, E. J.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Van Dam, H. J. J.; Wang, D.; Nieplocha, J.; Apra, E.; Windus, T. L., NWChem: A Comprehensive and Scalable Open-Source Solution for Large Scale Molecular Simulations. *Comput. Phys. Commun.* **2010**, 181, 1477-1489.

40. Kong, J.; White, C. A.; Krylov, A. I.; Sherrill, D.; Adamson, R. D.; Furlani, T. R.; Lee, M. S.; Lee, A. M.; Gwaltney, S. R.; Adams, T. R., Q‐Chem 2.0: A High‐Performance Ab Initio Electronic Structure Program Package. *J. Comput. Chem.* **2000**, 21, 1532-1548.

41. Schmidt, M. W.; Baldridge, K. K.; Boatz, J. A.; Elbert, S. T.; Gordon, M. S.; Jensen, J. H.; Koseki, S.; Matsunaga, N.; Nguyen, K. A.; Su, S., General Atomic and Molecular Electronic Structure System. *J. Comput. Chem.* **2004**, 14, 1347-1363.

42. *Jaguar*, Version 7.5; Schrödinger, L.L.C.: New York, NY, 2008.

43. *Vienna Ab Initio Simulation Package (VASP)*, Version 5.3.3; 2012.

44. Humphrey, W.; Dalke, A.; Schulten, K., VMD: Visual Molecular Dynamics. *J. Mol. Graphics* **1996**, 14, 33-38.

45. Pettersen, E. F.; Goddard, T. D.; Huang, C. C.; Couch, G. S.; Greenblatt, D. M.; Meng, E. C.; Ferrin, T. E., UCSF Chimera—A Visualization System for Exploratory Research and Analysis. *J. Comput. Chem.* **2004**, 25, 1605-1612.

46. Roe, D. R.; Cheatham III, T. E., PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data. *J. Chem. Theory Comput.* **2013**.

47. Michaud‐Agrawal, N.; Denning, E. J.; Woolf, T. B.; Beckstein, O., MDAnalysis: A Toolkit for the Analysis of Molecular Dynamics Simulations. *J. Comput. Chem.* **2011**, 32, 2319-2327.

48. Guha, R.; Howard, M. T.; Hutchison, G. R.; Murray-Rust, P.; Rzepa, H.; Steinbeck, C.; Wegner, J.; Willighagen, E. L., The Blue Obelisk-Interoperability in Chemical Informatics. *J. Chem. Inf. Model.* **2006**, 46, 991-8.

49. Murray-Rust, P.; Rzepa, H. S., Chemical Markup, XML, and the World Wide Web. 4. CML Schema. *J. Chem. Inf. Comput. Sci.* **2003**, 43, 757-72.

50. Phadungsukanan, W.; Kraft, M.; Townsend, J. A.; Murray-Rust, P., The Semantics of Chemical Markup Language (CML) for Computational Chemistry : CompChem. *J. Cheminform.* **2012**, 4, 15.

51. O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R., Open Babel: An Open Chemical Toolbox. *J. Cheminform.* **2011**, 3, 33.

52. de Jong, W. A.; Walker, A. M.; Hanwell, M. D., From Data to Analysis: Linking NWChem and Avogadro with the Syntax and Semantics of Chemical Markup Language. *J. Cheminform.* **2013**, 5, 25.

53. Stone, J. E.; Hardy, D. J.; Ufimtsev, I. S.; Schulten, K., GPU-Accelerated Molecular Modeling Coming of Age. *J. Mol. Graphics Modell.* **2010**, 29, 116-125.

54. Susukita, R.; Ebisuzaki, T.; Elmegreen, B. G.; Furusawa, H.; Kato, K.; Kawai, A.; Kobayashi, Y.; Koishi, T.; McNiven, G. D.; Narumi, T., Hardware Accelerator for Molecular Dynamics: MDGRAPE-2. *Comput. Phys. Commun.* **2003**, 155, 115-131.

55. Narumi, T.; Ohno, Y.; Okimoto, N.; Suenaga, A.; Yanai, R.; Taiji, M. A High-Speed Special-Purpose Computer for Molecular Dynamics Simulations: MDGRAPE-3. In NIC Workshop, 2006; 2006; Vol. 34; pp 29-36.

56. Shaw, D. E.; Dror, R. O.; Salmon, J. K.; Grossman, J. P.; Mackenzie, K. M.; Bank, J. A.; Young, C.; Deneroff, M. M.; Batson, B.; Bowers, K. J.; Chow, E.; Eastwood, M. P.; Ierardi, D. J.; Klepeis, J. L.; Kuskin, J. S.; Larson, R. H.; Lindorff-Larsen, K.; Maragakis, P.; Moraes, M. A.; Piana, S.; Shan, Y.; Towles, B., In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*; ACM: Portland, Oregon, 2009, pp 1-11.

57. Meyer, T.; Ferrer-Costa, C.; Pérez, A.; Rueda, M.; Bidon-Chanal, A.; Luque, F. J.; Laughton, C. A.; Orozco, M., Essential Dynamics: A Tool for Efficient Trajectory Compression and Management. *J. Chem. Theory Comput.* **2006**, 2, 251-258.

58. Omeltchenko, A.; Campbell, T. J.; Kalia, R. K.; Liu, X.; Nakano, A.; Vashishta, P., Scalable I/O of Large-Scale Molecular Dynamics Simulations: A Data-Compression Algorithm. *Comput. Phys. Commun.* **2000**, 131, 78-85.

59. Bernstein, F. C.; Koetzle, T. F.; Williams, G. J. B.; Meyer, E. F.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; Tasumi, M., The Protein Data Bank. *Eur. J. Biochem.* **2008**, 80, 319-324.

60. Allen, F. H.; Taylor, R., Research Applications of the Cambridge Structural Database (CSD). *Chem. Soc. Rev.* **2004**, 33, 463-475.

61. Wang, Y.; Xiao, J.; Suzek, T. O.; Zhang, J.; Wang, J.; Bryant, S. H., PubChem: A Public Information System for Analyzing Bioactivities of Small Molecules. *Nucleic Acids Res.* **2009**, 37, W623-W633.

62. Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B., ChEMBL: a Large-Scale Bioactivity Database for Drug Discovery. *Nucleic Acids Res.* **2012**, 40, D1100-D1107.

63. Willighagen, E. L.; Waagmeester, A.; Spjuth, O.; Ansell, P.; Williams, A. J.; Tkachenko, V.; Hastings, J.; Chen, B.; Wild, D. J., The ChEMBL Database as Linked Open Data. *J. Cheminform.* **2013**, 5, 1-12.

64. Pence, H. E.; Williams, A., ChemSpider: An Online Chemical Information Resource. *Journal of Chemical Education* **2010**, 87, 1123-1124.

65. Ng, M. H.; Johnston, S.; Wu, B.; Murdock, S. E.; Tai, K.; Fangohr, H.; Cox, S. J.; Essex, J. W.; Sansom, M. S. P.; Jeffreys, P., BioSimGrid: Grid-Enabled Biomolecular Simulation Data Storage and Analysis. *Future Gener. Comp. Sy.* **2006**, 22, 657-664.

66. Tai, K.; Murdock, S.; Wu, B.; Ng, M. H.; Johnston, S.; Fangohr, H.; Cox, S. J.; Jeffreys, P.; Essex, J. W.; Sansom, M. S., BioSimGrid: Towards a Worldwide Repository for Biomolecular Simulations. *Organic & biomolecular chemistry* **2004**, 2, 3219-21.

67. Simms, A. M.; Toofanny, R. D.; Kehl, C.; Benson, N. C.; Daggett, V., Dynameomics: Design of a Computational Lab Workflow and Scientific Data Repository for Protein Simulations. *Protein Eng. Des. Sel.* **2008**, 21, 369-377.

68. Toofanny, R. D.; Simms, A. M.; Beck, D. A.; Daggett, V., Implementation of 3D Spatial Indexing and Compression in a Large-Scale Molecular Dynamics Simulation Database for Rapid Atomic Contact Detection. *BMC Bioinformatics* **2011**, 12, 334.

69. Calleja, M.; Bruin, R.; Tucker, M. G.; Dove, M. T.; Tyer, R.; Blanshard, L.; Van Dam, K. K.; Allan, R. J.; Chapman, C.; Emmerich, W., Collaborative Grid Infrastructure for Molecular Simulations: The eMinerals Minigrid as a Prototype Integrated Compute and Data Grid. *Molecular Simulation* **2005**, 31, 303-313.

70. Baru, C.; Moore, R.; Rajasekar, A.; Wan, M. The SDSC Storage Resource Broker. In Proceedings of the 1998 Conference of the Centre for Advanced Studies on Collaborative research, 1998; IBM Press: 1998; p 5.

71. Thain, D.; Tannenbaum, T.; Livny, M., Condor and the Grid. *Grid computing: Making the global infrastructure a reality* **2003**, 299-335.

72. Alfredsson, M. eMinerals: Science Outcomes Enabled by New Grid Tools. In Proc. UK eScience All Hands Meeting, 2005; 2005; pp 788-795.

73. Meyer, T.; D'Abramo, M.; Hospital, A.; Rueda, M.; Ferrer-Costa, C.; Perez, A.; Carrillo, O.; Camps, J.; Fenollosa, C.; Repchevsky, D.; Lluis Gelpi, J.; Orozco, M., MoDEL (Molecular Dynamics Extended Library): A Database of Atomistic Molecular Dynamics Trajectories. *Structure* **2010**, 18, 1399-1409.

74. Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Lluis Gelpi, J., MDWeb and MDMoby: An Integrated Web-Based Platform for Molecular Dynamics Simulations. *Bioinformatics* **2012**, 28, 1278-1279.

75. Hospital, A.; Faustino, I.; Collepardo-Guevara, R.; Gonzalez, C.; Gelpi, J. L.; Orozco, M., NAFlex: A Web Server for the Study of Nucleic Acid Flexibility. *Nucleic Acids Res.* **2013**, 41, W47-55.

76. Vohra, S.; Hall, B. A.; Holdbrook, D. A.; Khalid, S.; Biggin, P. C., Bookshelf: A Simple Curation System for the Storage of Biomolecular Simulation Data. *Database: the Journal of Biological Databases and Curation* **2010**.

CHAPTER 3

DATA MODEL, DICTIONARIES, AND DESIDERATA FOR BIOMOLECULAR SIMULATION DATA INDEXING AND SHARING[1]

Abstract

Background

Few environments have been developed or deployed to widely share biomolecular simulation data or to enable collaborative networks to facilitate data exploration and reuse. As the amount and complexity of data generated by these simulations are dramatically increasing and the methods are being more widely applied, the need for new tools to manage and share these data has become obvious. In this paper we present the results of a process aimed at assessing the needs of the community for data representation standards to guide the implementation of future repositories for biomolecular simulations.

Results

We introduce a list of common data elements, inspired by previous work, and updated according to feedback from the community collected through a survey and

[1] Reprinted with permission from Thibault, J. C., Roe, D. R., Facelli, J. C., & Cheatham, T. E. (2014). Data model, dictionaries, and desiderata for biomolecular simulation data indexing and sharing. Journal of Cheminformatics, 6(1), 4.

personal interviews. These data elements integrate the concepts for multiple types of computational methods, including quantum chemistry and molecular dynamics. The identified core data elements were organized into a logical model to guide the design of new databases and application programming interfaces. Finally a set of dictionaries was implemented to be used via SQL queries or locally via a Java API built upon the Apache Lucene text-search engine.

Conclusions

The model and its associated dictionaries provide a simple yet rich representation of the concepts related to biomolecular simulations, which should guide future developments of repositories and more complex terminologies and ontologies. The model still remains extensible through the decomposition of virtual experiments into tasks and parameter sets, and via the use of extended attributes. The benefits of a common logical model for biomolecular simulations was illustrated through various use cases, including data storage, indexing, and presentation. All the models and dictionaries introduced in this paper are available for download at http://ibiomes.chpc.utah.edu/mediawiki/index.php/Downloads.

Background

Introduction

Thanks to a dramatic increase in computational power, the field of biomolecular simulation has been able to generate more and more data. While the use of quantum mechanics (QM) is still limited to the modelling of small biomolecules[1] composed of less than a couple hundred of atoms, atomistic or coarser-grain molecular representations have allowed researchers to simulate large biomolecular systems (i.e., with hundreds of thousands of atoms) on time scales that are biologically significant (e.g., millisecond for protein folding).[2] Classical molecular dynamics (MD) and hybrid approaches such as quantum mechanics/molecular mechanics (QM/MM) are some of the most popular methods to simulate biomolecular systems. With the explosion of data created by these simulations — generating terabytes of atomistic trajectories — it is increasingly more difficult for researchers to manage their data. Moreover results of these simulations are now becoming of interest to bench scientists to aid in the interpretation of increasingly complex experiments and to other simulators for assessing force fields and to develop coarse-grain models. Opening these large data sources to the community, or at least within collaborative networks, will facilitate the comparison of results to detect and correct issues with the methods, identify biologically relevant patterns or anomalies, and provide insight for new experiments. While the Protein Data Bank[3] is very useful as a central repository for structural data, the number of repositories for biomolecular simulations is still very limited. To the best of our knowledge the only databases that currently provide access to MD data for the community are Dynameomics[4, 5] and MoDEL (Molecular Dynamics Extended Library[6]). Dynameomics and MoDEL were populated with about 11,000 and 17,000 MD trajectories of proteins, respectively. One of the problems with such repositories is that the published data were generated in a specialized environment to study a given biological process (e.g., protein folding), resulting in fairly homogeneous data compared to the range of methods and software available to the community. These repositories are somewhat tied to these environments and it is uncertain how one would publish data generated outside these environments or how external systems would index or interface with these repositories. As more repositories are created the need for a common representation of the data becomes crucial to achieve semantic interoperability and enable the development of federated querying tools and scientific gateways. Note that other efforts to build repositories and scientific gateways, such as the BioSimGrid project[7] and work by Terstyanszky et al.,[8] have been undertaken but so far none has been widely adopted outside their original deploying institution or organization.

In the computational quantum chemistry community, more progress has been achieved towards the development of repositories using standard data representations to enable collaborative networks. One of the main on-going efforts is led by the Quixote project[9] which aims to create a federated infrastructure for quantum chemistry calculations where data is represented with CML CompChem (Chemical Markup Language – Computational chemistry[10]) and integrated into the semantic web through RDF (Resource Description Framework, http://www.w3.org/RDF/). The Chemical Markup Language[11] (CML) and its computational component CML-CompChem aim to provide a standard representation of computational chemistry data. While the core CML XML specifies the requirements to represent molecular system topologies and properties, CML-CompChem supplements CML to allow the representation of computational chemistry data, including input parameters and output data (calculations). So far these extensions have mainly focused on representing quantum computational chemistry experiments as XML files. These files can be created by converting input and/or output files generated by a particular software package through file parsers such as the ones supported by the Blue Obelisk group[12] (e.g., Chemistry Development Kit, Open Babel).

While CML-CompChem has a great potential for QM calculations,[13] its usefulness for MD and biomolecular simulations in general might be limited. For example, typically trajectories of atomic positions need to be compressed or binary encoded for data movement, storage purposes, and/or accuracy. Embedding this information into a verbose XML file such as CML will not be the optimal solution, at least not for the description and formatting of the raw output. Another obstacle to the conversion of MD experiments to a single-file representation is the common definition of many separate input files (e.g., system topology, method parameters, force field) necessary to prepare an MD simulation and define the different iteration cycles (e.g., minimization, equilibration, production MD). In quantum chemistry, the targeted molecules and calculation parameters are typically defined in a single input file (e.g., ".com" file for Gaussian[14] and ".nw" file for NWChem[15]) which makes this conversion much simpler. The output files generated by quantum chemistry software packages usually already contain the final results the user is interested in while in MD the raw output, i.e., multiple files containing the trajectories of atomic positions, energies and other output information, has to be further processed through various analysis tasks to create meaningful information. These postprocessing steps involve the creation of new input and output files, making the conversion of an experiment to a single XML file even more difficult.

Perhaps one of the main barriers to build repositories for biomolecular simulations is the lack of standard models to represent these simulations. To the authors' knowledge no published study has assessed the needs of the community regarding biomolecular simulation repository data models. Therefore it is unclear which pieces of information are considered essential by researchers and how they should be organized in a computable manner, so that users can:

- Index their data and build structured queries to find simulations or calculations of interest, not only via the annotations, but also with access to the raw data (files).
- Summarize, present, and visualize simulation data either through a web portal or more static documents (e.g., PDF document, XML file).

These models should be designed to include not only the description of the various independent computational tasks performed but also a high-level description of the overall simulated experiment. Each experiment can be related to multiple concepts that help understanding what was simulated, how, and in which context. These concepts can be grouped into the following categories:

- *Authorship*: information about the author, grants and publications related to the experiment
- *Methods*: computational method description (e.g., model building, equilibration procedure, production runs, enhanced sampling methodology) and associated inputs / parameters
- *Molecular system*: description of the simulated molecules from a structural, chemical, and biological point of view
- *Computational platform*: description of the software used to run the computational tasks, the host machine (computational environment), and execution configuration
- *Analysis*: derived data that can be used for quality assessment of the simulations

- *Files*: information about the raw simulation input and output files, such as format, size, location, and hosting file system

In this study we describe our efforts to formalize the needs of the community regarding the elements necessary to index simulation data. This work was initiated in part to support the iBIOMES (Integrated BIOMolEcular Simulations) project[16], an effort to create a searchable repository for biomolecular simulations, where the raw data (input and output files) is made available so that researchers can rerun the simulations or calculations, or reuse the output to perform their own analysis. In the initial prototype a set of software-specific file parsers were developed to automatically extract common data elements (metadata) and publish the raw data (i.e., the input and output files) to a distributed file system using iRODS[17] (integrated Rule-Oriented Data System). The published files and collection of files (experiments) are indexed based on the extracted data elements and are stored as attribute-value-unit triplets in a relational database. In this paper we introduce a list of common data elements and a data model that will help iBIOMES and future biomolecular simulation data repository developments move towards semantic interoperability.

Motivation for a common data representation: examples

The development of a common framework for data representation provides users with a large amount of flexibility to develop new tools for managing the data while maintaining interoperability with external resources. In this section we present three different examples that demonstrate the need for a standard representation of biomolecular simulation data, whether it is for indexing or presentation to the user. All three examples have been implemented to some extent in prototype form here. The first example is based on our experience with iBIOMES,[16] where simulation-specific metadata are associated at the file or directory level, through a specialized file system (iRODS[17]). The second example shows how one would use a model-based approach to build a repository where simulation parameters and provenance metadata are stored in a relational database. Finally the last example illustrates how a model-based API (Application Programming Interface) can be used to automatically generate XML and HTML summaries for the simulations being published.

Example 1: building a repository based on file annotations

One of the simplest ways to index simulations is to tag the associated files and directories with user annotations summarizing their content. These tags can be simply stored in a database or indexed via dedicated systems such as MapReduce[18, 19] or Apache Lucene.[20] This approach is well suited for fast searches based on keywords or attribute-value pairs. In the iBIOMES system[16] these tags are managed by the iRODS framework,[17] which enables the assignment of attribute-value-unit triplets to each file and directory in a distributed file system. This approach is very flexible since it allows the use of tags that represent common concepts such as computational methods and biological features, and user- or lab-specific attributes as well. In iBIOMES, a catalogue of common attributes was defined for users to annotate their data. The definition of such attributes is important as they can be tied to actionable processes, such as analyses, visualizations, and ultimately more complex workflows. It is then possible to build a user interface that presents the data and performs certain actions based on the existence of certain attributes or their associated values. For example if the format of a file is PDB (File format = “PDB”), then the user interface could enable 3D rendering of the associated molecules through Jmol.[21] A data dictionary that would offer possible values for a particular attribute is important as well. Each term should be well defined to leave no ambiguity to the user. A dictionary of force fields, for example, could list all the common force fields with a textual description, a type (e.g., classical, polarizable, coarse-grained), and the associated citations for each entry. A catalogue of common data elements, associated to a data dictionary, is also useful for users to pick from to facilitate annotations and build queries. The catalogue used in iBIOMES was defined internally by our lab and probably is not yet sufficiently exhaustive for the community at large. However, creating a catalogue of common data elements (CDE) supported by the community is a first step towards the standardization of biomolecular simulation data description. Defining a subset as recommended (i.e., the core data elements) would go a step further and set a criterion to assess the quality of the data publication process. Finally, linking these CDEs to existing terminologies or ontologies would bring semantic meaning to the annotations, enabling data discovery and query via external systems.

Example 2: building a repository based on a relational database

While a CDE catalogue is important, it lacks the representation of relationships between elements unless it is linked to a well-structured taxonomy. For example, if a user is interested in simulations of nucleic acids, a hierarchical representation of biomolecules could be used to infer that the user is actually looking for any simulation of DNA or RNA. The aim of a logical data model is to give a representation of the domain that captures the business needs and constraints while being independent from any implementation concern.[22] Such a model can provide the foundations for the design of a database and can be used to automatically generate API skeletons using modern modelling tools (e.g., Enterprise Architect, ArgoUML, Visual Paradigm). Since it is a domain-specific representation of the data, it can also serve as a starting point to develop a terminology or ontology specific to this domain. In this second example we demonstrate how a data model could be used to prototype a repository for biomolecular simulations where simulation parameters and provenance metadata are organized and stored in a relational database. We created a UML (Unified Modeling Language, http://www.uml.org/) model including logical and physical entities to build a relational database that could eventually be wrapped as a Grid service. The Grid[23] represents a great infrastructure for collaboration because of the underlying authentication scheme and data discovery services available, but also because of the semantic and syntactic integration. For this example we decided to mock up a data grid service using the caGrid[24] framework. caGrid was supported by the National Cancer Institute (NCI) and aimed to create a collaborative network for researchers to share cancer data, including experimental and computational data. The caCORE (cancer Common Ontologic Representation Environment) tools that were developed in this context facilitate the creation of the grid interfaces by automatically generating the necessary Java code from a UML model. These tools are now maintained by the National Cancer Informatics Program (NCIP) and available at: https://github.com/NCIP/. For this example we mapped the logical model to a data model using the caAdapter graphical tool. The final UML model and database creation scripts for MySQL (http://www.mysql.com/) are available for download at: http://ibiomes.chpc.utah.edu/mediawiki/index.php/Downloads. More details about the UML model are provided in the section introducing the logical data model. The caCORE SDK (Software Development Kit) was then used to generate the Hibernate (http://www.hibernate.org/) interfaces to the database along with a web interface that can be used to create simple queries or browse the published data. A screenshot of the generated interface is given in Figure 3.1 (listing of various published computational tasks). To actually build and deploy the data service onto a Grid, one would have to use the Introduce module. Semantic integration is also possible via the Semantic Integration Workbench (SIW), which enables tagging of the domain model with concepts from standard terminologies (e.g., ChEBI, Gene Ontology).

Example 3: representing experiments using XML

While a database provides a single endpoint to query data, other types of data descriptors become necessary when moving data between file systems, or simply to provide a light-weight description of the data. XML has been widely adopted by the scientific community to represent structured data because of its flexibility and support by web technologies. In the field of computational chemistry CML-CompChem[10] aims to provide a detailed representation of computations but currently lacks support in the molecular dynamics community. BioSimML[25] (Biomolecular Simulation Markup Language) was developed specifically for biomolecular modelling and supports QM/MM simulation representations but its current status is uncertain. The Unified Molecular Modeling (UMM) XML schema[26] is currently being developed by ScalaLife (Scalable Software for Life Sciences, http://www.scalalife.eu/) and will attempt to provide a detailed description of MD runs, so that these files can be used as a standard input to run within various MD engines. So far these XML-based formats have focused on giving a low-level representation of the simulation runs so that data can be converted between legacy formats. In this example we generate an XML-based representation of the experiment as a whole (multiple tasks), with a limited granularity for the description of each task. For this purpose we developed a Java API based on our logical model to generate XML representations of experiments (Figure 3.2). Format-specific file parsers developed for the iBIOMES project[16] read in input and output files associated to an experiment to create an internal representation of the experiment and associated computational tasks. In the Java code, classes are annotated with Java Architecture for XML Binding (JAXB, https://jaxb.java.net/) annotations to map the logical model to an XML schema. The JAXB API can then be used to automatically output XML documents based on the internal Java representation of the experiment or read in an XML file to build the Java objects. The same process could be implemented in various languages, using CodeSynthesis XSD (http://www.codesynthesis.com/products/xsd/) in C++ or PyXB (http://pyxb.sourceforge.net/) in Python for example.

The XML output does not aim to be sufficient to recreate input or output files in legacy formats but it will provide enough information for users to rapidly understand the computational methods and structures represented by the associated raw data. This type of XML document can be used as a way to give a detailed summary of experiments when exchanging data, compressed with the raw data for example. These documents can be transformed through XSLT (eXtensible Stylesheet Language Transformations) to be rendered as HTML pages and build repository web interfaces. A sample XML output along with an HTML-based tree view generated through XSLT are presented in Figure 3.3. For this example a set of AMBER-specific[27] file parsers was used to parse a directory containing all the input and output files associated to an MD study of RNA. Common data elements related to the molecular system topology were extracted from the AMBER parameter/topology file while task (minimization and MD runs), parameter set (e.g., implicit solvent, number of iterations), and computational platform information were extracted from the AMBER MD output files.

Summary

These three prototypes serve as examples demonstrating the need for a catalogue of CDEs and the representation of relationships between concepts through a data model. The catalogue of CDEs, associated to a data dictionary, provides the basis for a controlled vocabulary that can be used to annotate experiment data (e.g., files and directories) and build queries. The data model provides extra information as it links concepts together and allows more complex and structured queries, through a relational database, for example. The second example showed how modern software engineering tools can use data models to generate database schemas and APIs for repository developments. Finally the last example showed that XML representations can be easily generated if the API follows a model-based approach.

In this paper we introduce a list of CDEs built upon community feedback, and a logical model that ties dictionaries and common data elements together. Common data elements for simulation data indexing and presentation were identified through a survey, while recommendations are made for trajectory and analysis data description. The common data elements were organized through a logical data model, which was refined to include dictionaries and minimize data redundancy. Finally the design and implementation for a subset of these dictionaries are introduced.

Experimental

Identification of core data elements

Survey

A survey was distributed to the community to assess the list of data elements that was defined in iBIOMES[16]. This initial list of common data elements was based on the BioSimGrid[7] data model and supplemented with new elements to reflect the needs of our lab and various collaborators at the University of Utah, and to add descriptions of quantum chemistry calculations. The main goal of the survey was to identify which elements were missing and which ones were not so important according to the community. A list of 47 data elements describing simulation runs and the associated files was presented to experts. These data elements were grouped into 6 categories for organizational purpose: authorship (user information and referenced citations related to a particular run), platform (hardware/software), molecular system (molecules being studied, independently from the model chosen), molecules (info about the molecules composing the system), methods (can apply to any method, including QM and MD), molecular dynamics, and quantum mechanics. The experts were asked to score the data elements based on how important they are to them to describe their own data and/or to index community data and build search queries. Scoring was based on a Likert scale (1 = “Not important at all”, 2 = “Not very important”, 3 = “Not sure”, 4 = “Important”, 5 =

"Very important", and "N/A" for nonapplicable). In each group, the experts were also allowed to propose missing data elements and/or comment on the listed elements.

The survey was made available online (see extract in Appendix A) in March 2012 for about a month and promoted through the Computational Chemistry List (CCL) and the AMBER developers' mailing list. The CCL list is a fairly well known group for general discussions related to computational chemistry, perhaps with an emphasis on QM-related methods. The AMBER developers group represents a variety of theoretical disciplines (MD, QM, QM/MM), with developments targeting various types of systems (e.g., proteins, nucleic acids, lipids, carbohydrates, small compounds) and discussions on how to best use the software, methods and force fields. Individual emails were also sent to different research groups at the University of Utah that are specialized in computational chemistry.

Trajectory and analysis data

The survey did not include any analysis- or file-related data elements. The Dublin Core metadata (http://dublincore.org/documents/dces/) can be used as a good reference to describe files at a high level (e.g., author, format). Analysis data on the other hand is very complex to describe because of its direct relation to the raw data it derives from (e.g., use of multiple input files representing experimental and computed data) and the existence of numerous analysis methods that can be problem-specific (e.g., Protein vs. RNA, QM vs. MD). In most cases it will not make sense to use analysis data to index an experiment either. For example looking for MD trajectories with a particular RMSD (root mean square deviation) value would be irrelevant without providing more context about the system and the method used to calculate the value. Although analysis data is a key factor to assess the quality of a simulation, its use for data indexing and retrieval is not trivial and therefore was not included in the survey. A generic framework for the description of trajectory and derived data is nevertheless provided in the Results section.

Logical model

Overview

The logical model presented here was derived from a conceptual model that organized all the identified common data elements into a defined domain. The conceptual model was reduced into a logical model with the assumption that the raw input and output files are made available (in a repository similar to iBIOMES or MoDEL) and that the model would be used to index the data rather than providing a complete view of the results (e.g., calculation output, structures defined in each MD trajectory frame). Although analysis data and quality criteria are crucial to provide an objective perspective on experiment results, no associated concept was included in the current model. The granularity of the model was limited to a sufficient level of details that makes it computable. For example, the description of the theory behind modelling methods is not part of the model. The end-goal being to share the results of the simulations or calculations with the community, we limited our model to include only popular methods that are used for the study of biomolecules or smaller ligands.

Use of dictionaries

One of the main features of this logical model is the integration of dictionaries to avoid data redundancy. For example a dictionary containing definitions of force fields (e.g., name, type, citations) can be referenced by molecular dynamics tasks, instead of creating individual force field definition entries every time the force field is used. The integration of dictionaries into the model should not enforce mappings to standard definitions but rather enable links between specific values and standard definitions only if they exist. If no mapping exists the user should still be able to publish the data. This is achieved through the storage of “specific names” outside the dictionaries with an optional reference to the term definition, where the standard version of the name (not necessarily different) is defined. For example if the basis set “LANL2DZ” is used in a QM calculation, but no corresponding entry exists in the basis set dictionary, the name of the basis set will still be stored in the database when publishing the data to allow queries on the calculation.

Units

Certain attributes need to be associated to a unit to be understood by a human or a computer. Different software packages might use different units to represent the same attribute. For example, distances in AMBER[27] are measured in Ångströms while GROMACS[28] uses nanometres. When publishing data to a repository one should either convert the values using units previously agreed upon or make sure that the units are published along with the values. In both cases, mechanisms should be in place to provide a description of the units when pulling data from the repository. For the description of this model we assume that the units are already set in the repository. Therefore they are not included in the description of the model.

Dictionaries

While most of the data described in a logical model for biomolecular simulations can be directly parsed from the input and output files, dictionaries containing standard definitions and values for certain data elements need to be prepopulated. In this paper we present the design and implementation of several dictionaries that can be used to facilitate data publication and queries. For example, if a user is interested in QM calculations based on Configuration Interaction (CI) theory, a dictionary of all CI methods will be needed to return all the calculations of interest (e.g., CISD, CISD(T)). Another interesting use of these dictionaries is within the code of the file parsers. Instead of defining standard values within the code, one can use these dictionaries to look up information on the fly, and possibly use it to publish the data into the target repository.

An initial set of dictionaries was populated using the BiosimGrid[7] database dictionaries (source code available at: http://sourceforge.net/projects/biosimgrid/). They were then refined internally and supplemented with new dictionaries, especially to include QM-related definitions (e.g., basis sets, QM methods).

Results

Identification of core data elements

Survey

At the closing of the survey we were able to collect 39 responses (20 through CCL, 10 through the AMBER developers list, and 9 through emails). The results of the survey are presented in Appendix A. The respondents listed a few data elements they felt were missing from the proposed list or that needed to be refined (see comments in Appendix A). For instance, in the authorship category, a data element representing research grants was missing. For the representation of the molecular system, data elements representing important functional groups of the solute molecules should be added, along with, optionally, the apparent pH of the solvent. Adjustments should also be made to distinguish the different species in the system and flag them as part of the solvent or the solute. For the computing environment information, a respondent showed interest in knowing whether the software package is compiled in single, double, or mixed precision, what the memory requirements are for a run, and even what parallelization scheme is used. All these elements are very technical and might interest only a very limited number of users, even in the developer's community. The notion of hardware architecture was not clearly defined in the survey since it should have already included the use of GPU (see comment in Appendix A). A better representation of the hardware architecture can be done through three different data elements: the CPU architecture (e.g., x86, PowerPC), the GPU or accelerator architecture (e.g., Nvidia GeForce GTX 780, AMD Radeon HD 7970, Intel PHI), and possibly a machine or supercomputer architecture identification (e.g., Cray XK7, IBM Blue Gene/Q, commodity Infiniband cluster, etc.) and name (stampede.tacc.utexas.edu, h2ologin.ncsa.illinois.edu, keeneland.gatech.xsede.org, etc.). For the computational methods, data elements were missing for the representation of both MD and QM-specific parameters. In QM, the following elements were missing: exchange-correlation functionals (for DFT), pseudopotentials and plane wave cut-offs, and whether frozen core calculations are performed or not. Some comments pointed out the fact that the notion of convergence can be very subjective, especially when dealing with MD trajectories where multiple minima (conformations) can be found over time (see comments in Appendix A). The convergence flag and criteria were assigned as QM-specific data elements to reflect this. For MD, the context of the run (i.e., whether it is a minimization, an equilibration, or a production run) was missing. Representations of restraints and advanced sampling methods (e.g., replica-exchange, umbrella sampling) were also missing. More detailed properties were listed by the respondents. These included the order of expansion for LINCS-based constraints and the order of interpolation for Particle-Mesh Ewald. At this point it is not clear if such parameters need to be tracked since users would hardly use these to create queries and we assume that they can be directly read from the raw input files if necessary.

Based on the results of the survey and the various comments of the community we propose a set of common data elements for biomolecular simulation data indexing, listed in Appendix A. The identified elements were reorganized by making a distinction between data elements (concepts) and attributes (properties). For example the barostat data element has at least one property: an implementation name (e.g., Andersen, Berendsen). Depending on the type of barostat other properties could include a time constant and a chain length (e.g., Nose-Hoover barostat). We also included “derived” properties that would be inferred from other properties if the right terminology or dictionary is available. For example, the name of a QM method (e.g., MP2, B3LYP) should be enough to infer the level of theory (e.g., Møller-Plesset, DFT), and the name of the force field (e.g., AMBER FF99SB) should be sufficient to infer its type (e.g., classical). This distinction is important as it can help the developers choose which properties should be actually stored (e.g., in a database or an XML file) and which ones could be inferred. The set also contains recommended and optional data elements/attributes. An attribute is marked as recommended if its average score (i.e., the sum of Likert scale scores divided by the number of responses for that element) is greater than 4.0 ("Important"). Otherwise it is marked as optional. Attributes proposed by the respondents were categorized through an internal review performed by our lab, composed of researchers running molecular dynamics simulations and quantum chemistry calculations on a daily basis. A data element is considered recommended if it has at least one recommended attribute. The current list contains 32 data elements and 72 attributes (including 30 recommended attributes).

We recognize that the process by which the data elements were defined and characterized is not perfect. Although the number of respondents was fair (between 37 and 39 depending on the data element), certain data elements had to be added or redefined based on an internal review by some of our lab members, which might have created some bias towards the needs of our lab rather than a general consensus in the community. Despite these limitations the list of data elements proposed here may be considered the first attempt to summarize the needs of the computational chemistry community to enable biomolecular simulation data indexing and queries. This list should be a good starting point to create a list of standard metadata to tag files using simple attribute-value pairs or attribute-value-unit triplets, as is the case for iBIOMES via the iRODS metadata catalogue.[17] Although this list is fairly exhaustive, it is not complete and we hope that by publishing it the community will be able to provide more feedback and build on it, with the intent of this data model being extensible. The list is available on the iBIOMES Wiki at: http://ibiomes.chpc.utah.edu/mediawiki/index.php/Data_elements. Field experts who want to contribute to the list can request an account on the wiki.

Trajectory files

In most MD software packages the computed trajectories of atomic coordinates are stored in large files (~MB-TB) with each containing one or multiple time frames (e.g., PDB, AMBER NetCDF, DCD). This is the raw data that repositories would actually store and index for retrieval. Until now we have been focusing on the description of the computational tasks that were used to generate these data, i.e., the provenance metadata. These metadata can be used to find a given experiment and all associated trajectory files. On the other hand new attributes need to be assigned at the trajectory file level to describe their content and ultimately enable automatic data extraction and processing by external tools (e.g., VMD,[29] CPPTRAJ,[30] MDAnalysis[31]). Such attributes include the number of time frames, time between frames, number of atoms in the system and/or reference to the associated topology file, presence or absence of box coordinates, velocity information, and so on. It is important to note that the use of self-descriptive formats such as NetCDF (http://www.unidata.ucar.edu/software/netcdf/) would allow trajectory files to carry not only the description of the dataset, but also the provenance metadata, for example using the CDEs previously defined. Perhaps one of the most important attributes to give context within a full experiment is the index of a trajectory file within the set of all trajectory files representing a given task or series of tasks. Although self-descriptive formats could easily keep track of this information, it is nontrivial to generate such an index as tasks can be run independently outside of a managed workflow such as

MDWeb,[32] which would be able to assign these indexes at file creation time. The order of trajectory files is therefore commonly inferred from their names (e.g., “1.traj, 2.traj, 3.traj”). This approach usually works well although some errors might occur when trying to automate this ordering process. For example “10.traj” would be ranked before “2.traj” if a straight string comparison is performed (vs. “02.traj”). Strict naming conventions for trajectory data (raw, averaged, and filtered on space or time) should help circumvent these problems.

Analysis data

Although some analysis tasks are common to most biomolecular systems for a particular method (e.g., RMSD calculations of each frame in the trajectory to a reference structure) the number of analysis calculations one can perform is virtually infinite. There is currently no standard to describe the output of the analysis. Some formats might enable the description of the values (e.g., simple CSV or tab-delimited file with labelled columns and/or rows) but more structured files are required to describe the actual analysis process that generated the set of values contained in the file. Formats such as NetCDF are adapted to store this kind of description but are not commonly used to store biomolecular simulation analysis data. Instead comma- or tab-delimited files formats are usually preferred for their simplicity, readability, and support by popular plotting tools (e.g., MS Excel, OpenOffice, XmGrace). Assuming that the dataset is physically stored in such a file or in a relational database, a minimal set of attributes should be defined to facilitate reproduction of the analysis, as well as enable reading and loading into visualization tools with minimal user input. We believe that the strategy used in the NetCDF framework to break down data into variables with associated dimensions is a simple and logical one, and so we follow a similar strategy here:

- Data dimensions: Defines dimension sizes for defined data sets (i.e., variables). Any number of dimensions (including zero if data are scalar) can be defined.
- Data variables: The actual data. Report type (e.g., integer, float), labels, and units for all the values contained in a given set. One or more dimensions can be associated with a given variable based on its overall dimensionality. Zero dimensions correspond to a single value (e.g., average RMSD value), one dimension is an array (e.g., RMSD time series), two dimensions are a matrix (e.g., coordinate covariance), etc.

Another set of attributes need to be defined to represent the provenance metadata, i.e., how the analysis data were derived from the raw trajectories. Although different analysis tasks will require different input data types and parameters, a list of common attributes can be defined to provide a high-level description of the analysis task:

- Name (e.g., "RMSD") and description ("Root mean square deviation calculation") of analysis method (see entries defined in our MD analysis method dictionary)
- Path to the input file describing the task (if applicable)
- Name and version of the program used, along with the actual command executed
- Execution timestamp
- Reference system, if any (self, experimental, or other simulated structure)

While these attributes might not be sufficient to automatically replicate the results they should provide enough information for users other than the publisher to understand how the analysis data were generated and how the analysis task can be replicated.

A further set of attributes can be defined to provide additional details on the scope of the analysis and describe in detail the data from which the current data have been derived:

- File dependencies
- Filter on time
- Filter on space (e.g., heavy atoms only, specific residue)

These would facilitate maximum reproducibility as well as enable detailed searches on very specific types of analysis. The 'File dependencies' attribute may include information like the trajectory used in a given calculation, which could also be used to check if the current analysis is up-to-date (e.g., if the trajectory file is newer than the analysis data, the analysis can be flagged as needing to be updated). The 'Filter on time' attribute might describe a specific time window or subset of frames used in the analysis. Since these attributes are perhaps not as straightforward for analysis programs to report as the other attributes, they could be considered optional and/or set by the user after the data are published. The 'Filter on space' attribute could be particularly useful, since it would allow one for example to search for all analyses of a particular system done using only protein backbone atoms or only heavy atoms, etc. However, this would require translation of each individual analysis program's atom selection syntax to some common representation, which is no small task and would increase the size of the metadata dramatically for certain atom selections. In many cases it is likely that the atoms used in the analysis could be inferred from the command used, so this attribute could also be considered optional. Two examples of how these attributes might be applied to common analysis data are given in Appendix B.

Logical model

Overview

In this model the central concept is the virtual experiment, a set of dependent computational tasks represented by several input and output files. The goal of this model is to help create a common description of these virtual experiments (stored in a database or distributed file system for example) for indexing and retrieval. The overall organization of virtual experiments is illustrated in Figure 3.4. For the rest of this paper virtual experiments will be simply denoted as experiments. The organization of an experiment as a list of processes and tasks was inspired by the CML-CompChem[10] schema. In CML-CompChem the job concept represents a computer simulation task and can be included into a series of consecutive subtasks designated as a job list. The concepts of experiment, process group, process, and task are introduced here to handle the representation of tasks that might be run in parallel or sequentially, and that might target the same or different systems. An experiment process group is defined as a set of computational processes targeting the same molecular system, where a process is defined as a set of similar tasks (e.g., minimization tasks, MD tasks, QM tasks). In MD, the minimization-heating-production steps can be considered as a single process group with 3 different process instances. If multiple copies of the system are simulated, each copy will be considered a separate process group. In QM, a process would represent a set of sequential calculations on a compound. If various parts of the overall system are studied separately (e.g., ligand vs. receptor), each subsystem should be assigned to a different process group.

Within the scope of an experiment, multiple tasks and group of tasks will be created sequentially or in parallel, and based on intermediate results. To keep track of this workflow, dependence relationships (dependencies) can be created between tasks, between processes, and between process groups.

Notations

In the following sections we present the overall organization of the model through an object-oriented approach where the concepts (e.g., experiments, tasks, parameter sets, and molecular systems) are represented by classes with attributes. The description is supported by several class diagrams using the UML notation. For example inheritance is characterized through a solid arrow with an unfilled head going from the child to the parent class. Along with standard UML notations, we defined the following colour scheme to guide the reader:

- Blue: classes giving a high-level description of the experiments and tasks
- Yellow/orange: method/parameter description
- Green: classes describing the molecular system independently from the computational methods
- Pink: classes related to authorship and publication (e.g., citations, grants)
- Grey: description of the hardware or software used to run the tasks

Finally, classes representing candidates for dictionary entries are marked with wider borders.

Experiments, processes, and tasks

Figure 3.5 presents the concepts that can be used to describe the context of an experiment. Each experiment can be given a role, i.e., the general rationale behind the experiment. Examples of experiment roles include simulation (dynamics), geometry optimization, and docking. These roles should not be associated to any computational method in particular. Each experiment can be linked to a particular author (including institution, and contact information) to allow collaborations between researchers with common interests. Publications related to a particular experiment (citations) or that use the results of the experiments can be referenced. Grant information is important as well since it allows researchers to keep track of what their funding actually supports.

Experiment sets (Figure 3.2) are collections of independent experiments that are logically associated together, because of similar context (e.g., study of the same system using different methods) or simply for presentation purpose or to ease retrieval by users (e.g., all the experiments created by a certain working group). An experiment can be assigned to multiple experiment sets.

An experiment task corresponds to a unique computational task defined in an input file. Figure 3.6 presents the main concepts associated to experiment tasks. These include the definition of the actual calculation (e.g., frequency calculation and/or geometry optimization in QM, whether the dynamics of the system are simulated), the description of the simulated conditions (reference pressure and temperature), and the definition of the method (e.g., QM, MD, minimization) and input parameters (e.g., basis set, force field). More details about the different types of tasks and simulation parameters are given in the computational method section. Each task is executed within a computing environment, i.e., the set of hardware and software components used to run the simulation software package. These components include the operating system, the processor architecture, and the machine/domain name. Information about the task execution within the computing environment, including execution time, start and end timestamps, and termination status can be tracked as well. The software information includes name (e.g., "AMBER") and version ("12"). In certain cases a more specific name for the executable is available. This can provide extra information about the compilation step and/or the features available. In Gaussian,[14] for example, this information can be found in the output files: "Gaussian 09" would give a generic version of the software package while "EM64L-G09RevC.01" would give the actual revision number ("C.01") and the target architecture of the executable (e.g., Intel EM64). For AMBER, the executable name would be either "SANDER" (Simulated Annealing with NMR-Derived Energy Restraints) or "PMEMD" (Particle-Mesh Ewald Molecular Dynamics), which are two alternatives to run MD tasks within the software package.

Computational methods

The most common methods for biomolecules include QM, MD, and hybrid QM/MM. In this model we focus on these methods but we allow the addition of other methods by associating each task to one or multiple parameter sets that can be combined to create new hybrid approaches. This decomposition was applied to MD, minimizations (e.g., steepest descent, conjugate gradient), QM, and QM/MM methods as illustrated in Figure 3.7.

Common attributes of any computational method are represented at the ExperimentTask level. These include names (e.g., “Molecular dynamics”), description (e.g., “new unknown method”), types of boundary conditions (periodic or not), and the type of solvent (in vacuo, implicit, or explicit). Method-specific tasks (MinimizationTask, MDTask, QMTask, QMMMTask) are created to capture the parameters that would not be shared between all methods. Simulation parameters include any parameter related to the method or task that would be set before a simulation is run. These parameters are aggregated into sets that can be reused between methods. For example, the MD-specific task (MDTask) references MDParameterSet, which includes the definitions of the barostat, thermostat and force fields. The QM/MM-specific task (QMMMTask) references the same parameter set since these definitions are necessary to describe the computational method to treat the MM region. It also references a QM-specific parameter set to describe the QM method and a QM/MM-specific parameter set to describe the treatment of the QM/MM boundary. A new task type could be created for multilevel quantum calculations. In this case the task would reference multiple QM parameter sets and a new type of parameter sets that would define at least the algorithm or implementation used to integrate the different levels (e.g., ONIOM[33]).

In molecular dynamics, the behaviour of the simulated system is governed by a force field: a parameterized mathematical function describing the potential energy of the system, and the parameters of the function, with dynamics propagated using Newton’s equations of motion and the atomic forces determined from the forces or first derivatives of the potential energy function. Different parameters will be used for different types of atoms (or group of atoms in the type of coarse grain dynamics). A given force field parameter set is usually adapted to particular types of residues in molecules (e.g., nucleobases in nucleic acids vs. amino acids in proteins). For a single molecular dynamics task multiple force fields and parameter sets can be used simultaneously. When simulating an explicit water-based solvent for example, the specific force field parameter set used to represent these water molecules (e.g., TIP3P, TIP4P, SPC/E[34]) will typically be different from the set used to parameterize the atoms of the solute or the ions. The ForceField class presented in Figure 3.8 represents instances of force fields referenced by a particular run while ForceFieldDefinition represents an entry from the dictionary listing known force fields. Force field types include classical, polarizable, and reactive force fields.

Molecular dynamics methods can be classified into more specific classes of methods. For example in stochastic dynamics (Brownian or Langevin Dynamics), extra parameters can be added to represent friction and noise.[35] In coarse-grain dynamics the force field is applied to groups of atoms rather than individual atoms. The differentiation between atomistic and coarse-grain dynamics is then achieved solely based on the type of force field used. In this model Langevin dynamics and coarse-grain dynamics are not represented by different types of tasks as they share the same parameter set as classic molecular dynamics. The collision frequency attribute used specifically by stochastic dynamics was added to the MD parameter set while a flag specifying whether the force field is atomistic or coarse grain is set in the force field dictionary.

Each parameter set can be associated to a barostat and a thermostat to define how pressure and temperature are constrained in the simulated system (Figure 3.8). The ensemble type (microcanonical, canonical, isothermal–isobaric, or generalized) can be defined directly in the parameter set. The model also includes the concepts of constraints and restraints. Both have a target (i.e., the list of atoms they apply to), which can be described by an atom mask or a textual description (e.g., ':WAT', 'water'). The type of constraint is defined by the algorithm used (e.g., SHAKE, LINCS) while the type of restraint is characterized by the property being restrained (e.g., bond, angle).

Enhanced sampling methods are gaining interest in the MD community as larger systems and longer time scales can be simulated faster than with classic approaches.[36] These methods usually involve the creation of multiple ensembles or replica that can be run in parallel (e.g., temperature replica-exchange, umbrella sampling). A dictionary of such methods was created to list popular enhanced sampling methods. At the core the runs based on these methods can still be represented with multiple molecular dynamics tasks. Depending on the method, the implementation, and the definition of the input files, the set of MD tasks corresponding to a given enhanced sampling run can be grouped into processes where each process represents either a separate ensemble/replica or a group of tasks run in parallel. For a replica exchange MD (REMD) run using 4 replicas, one could either group the 4 MD tasks into a single process representing the whole REMD run or 4 separate processes with a single task each.

In quantum chemistry the two main elements that define the theory and approximations made for a particular run are the level of theory (or QM method) and the basis set (Figure 3.9). Basis sets provide sets of wave functions to create molecular orbitals and can be categorized into plane wave basis sets or atomic basis sets. They are defined in a dictionary (BasisSetDefinition). Different levels of theory are available to approximate the selected basis set and find a discrete set of solutions to the Schrödinger equation. Popular methods include Hartree-Fock and post-Hartree-Fock methods (e.g., Configuration Interaction, Møller-Plesset, Coupled-Cluster), multireference methods, Density Functional Theory (DFT), and Quantum Monte Carlo.[37] The classification of QM methods is not trivial because of the range of features dependent on the level of theory. For example, DFT method names typically correspond to the name of the exchange-correlation functional while semiempirical method names provide a reference to the empirical approximations of the method. For this model we defined the concepts of QM method, class and family. At the highest level the family defines the method as ab initio, semiempirical, or empirical. The class defines the level of theory for ab initio methods (e.g., Hartree-Fock, Møller-Plesset, Configuration Interaction, DFT, Multireference), or the type of semiempirical method (pi-electron restricted or all valence electron restricted). Note that one method can be part of multiple classes (e.g., Multireference configuration interaction, hybrid methods). At the lowest level the method name (e.g., MP2, B3LYP, AM1) corresponds to a specific method, as it would be called by a particular software package. Approximations of pure ab initio quantum methods can be used to reduce the computational cost of the simulations. Typical approximations include the use of frozen cores to exclude inner shells from the correlation calculations and pseudopotentials (effective core potentials) to remove the need to use basis functions for the core electrons. The use of such approximations is noted at the QM parameter set level.

Molecular dynamics methods can be “improved” by injecting quantum characteristics to the models (semiclassical methods). In ab initio molecular dynamics, the forces for the system are calculated using full electronic structure calculations, avoiding the need to develop parameters a priori. In hybrid QM/MM, the simulation domain is divided into an MM space where the MD force field applies, and a QM space where molecular orbitals will be described. Different methods exist to treat the boundaries between the two spaces. The decomposition of runs into tasks and parameter sets make the integration of such methods possible and fairly straight forward. For example, one could create a new type of tasks for ab initio molecular dynamics that would have at least two parameter sets: the QM parameter set defined earlier and a new parameter specific to ab initio molecular dynamics that would define the time steps (number, length) and the type of method (e.g., Car-Parinello MD, Born-Oppenheimer MD).

Molecular system

In this model a distinction is made between biomolecules (e.g., RNA, protein) and "small molecules" (Figure 3.10). Here we define a small molecule as a chemical or small organic compound that could potentially be used as a ligand. They are defined at the level of a single molecule while biomolecules are described by chains of residues. Typically, QM calculations will target small molecules while MD simulations will target larger biomolecules and ligand-receptor complexes. Properties such as molecular weight and formula are worth being tracked for small compounds but their importance is not that obvious when dealing with larger molecules.

Three dictionaries are necessary to provide definitions for standard residues, atomic elements (as defined in the periodic table), and element families (e.g., Alkaline, Metals). Note that here we minimize the amount of structural data by keeping track of occurrences of residues (ResidueOccurrence) and atom types (AtomOccurrence) in a particular molecule, rather than storing individual instances. For example, in the case of water, there will be a single entry for the hydrogen atom with a count set to 2, and another entry for the oxygen atom with a count set to 1. The same approach is used to keep track of the various molecules in the system. For example explicit solvent using water would be represented by the definition of the water molecule and the count of these molecules in the system. To enable searches of specific ligands a simple text representation of the compound is necessary. Molecule identifiers such as SMILES (Simplified Molecular-Input Line-Entry System[38]) or InChI (International Chemical Identifier[39]) strings can be associated to small molecules to enable direct molecule matching and similarity and substructure searches. The residue sequence is also available to search biomolecules based on an ordered list of residues. The residue sequence can be represented by two different strings: the original chain, or specific chain, as referenced in the input file defining the molecular topology, and a normalized chain. The specific chain can potentially give more information about the individual residues within the context of the software that was used, and reference nonstandard residues defined by the user. The normalized chain on the other hand uses a normalized nomenclature for the residue: one-letter codes representing either amino-acids or nucleobases. The normalized chain can be used to query the related molecule without prior knowledge about the software used, and enables advanced matching queries (e.g., BLAST [40]).

Both residue and atom occurrences can be given a specific symbol, which represents a software-specific name, usually referencing a computational model for the entity. In MD the specific symbol would be the force field atom type while in QM this would be used to specify which basis set should be applied.

The description of the biomolecules should include at least a generic type such as DNA, RNA or protein to classify the simulated molecules at a high level. Other biological information such as species (e.g., Mus musculus, Homo sapiens) and molecule role can be added as well. As defined by the Chemical Entities of Biological Interest (ChEBI[41]), each molecule can have one or multiple roles (application, chemical role, and/or biological role). This data element is very important as it would allow researchers to query molecules based on their function rather than their structure. On the other hand this type of information is not included in the raw simulation files, which means that it would have to be entered manually by the owner of the data. To avoid this one can imagine populating this information automatically by referencing external databanks that already store these attributes (e.g., Protein Data Bank[3]). This is reflected in this model by the reference structure concept, which keeps track of the database and the structure entry ID. If the topology of a simulated system is actually derived from a reference structure an extra field can be used to describe the protocol used to prepare the reference structure so that it serves as an input of the simulations. Possible steps include choice of the specific model number if several are available in a single PDB entry or which PDB entry if multiple entries are possible, possible addition of missing residues from disordered regions, or specification of homology or other putative models.

Files and file system

So far the description of the model focused on the data elements related to the experiment itself to explain why the different tasks were run and what they represent. Another important aspect of this model is the inclusion of a reference to the files (input and output) that contain the actual data being described. This is illustrated in Figure 3.11. Each experiment can be associated to one or several file collections stored on local or remote file systems (e.g., NFS, Amazon S3, iRODS server). For each of these collections no assumption should be made on the location or the implementation of the file system. Therefore it is necessary to keep track of the type of file server and host information to find a route to the host and access the files using the right protocol and/or API. The individual files should be associated to the tasks they represent and a distinction between input (parameters and methods) and output (e.g., logs, trajectories) files should be made. The topology files should be associated to the molecular system instead. Note that in certain cases, especially for QM calculations, the topology and input parameters might be contained in the same file. Each file reference should at least contain a unique identifier (UID) within its host file system and a format specification.

Extended attributes

It is obvious that no single data model will be able to capture the needs of any lab running biomolecular simulations. The intent of this logical model is to provide a simple yet fairly exhaustive description of the concepts involved. To allow the addition of new properties, to provide more details about the experiment or to keep track of user- or lab-defined attributes, the notion of extended attribute can be introduced to the model. Each extended attribute would be an attribute-value-unit triplet referenced by a given class to extend its own attributes, as defined in the logical model. For example one user might want to keep track of the order of interpolation and the direct space tolerance for PME-based simulations. These parameters are currently not represented in the model, which only keeps track of the name of the electrostatics model (“PME”). To add these two parameters, one could add two extended attributes to the MD parameter set class (Figure 3.8) called “PME interpolation order” and “PME tolerance.”

From an object-oriented perspective, all the classes introduced in the logical model could inherit from a single superclass that would reference extended attributes, where each extended attribute would be an attribute-value-unit triplet with a possible link to a concept identifier defining the attribute in an existing terminology. From a database perspective, an extra table would be needed to store all the extended attributes. Such table would need the necessary columns to represent the attribute-value-unit triplet, a possible concept identifier, and the name of the table each attribute would extend. Although this is an easy way to gather all the extended attributes in a single table this approach is not rigorous from a relational approach. To allow SQL queries that do not involve injection of table names each table would have to be associated to an extra table storing its extended attributes.

Summary

The logical model presented here defines a domain that should be sufficient to index biomolecular simulation data at the experiment level. In total over 60 classes were defined to represent the common data elements identified through the survey, along with new elements and dictionaries that should avoid data redundancy and facilitate queries using standard values. From a developer’s perspective this model provides some guidelines for the creation of a physical data model that would be more dependent on a particular technology, whether it is for the implementation of a database or an API. At a more abstract level the concepts introduced in this logical model provide a good starting point for the creation of a new terminology or ontology specific to biomolecular simulations.

Dictionaries

Overview

The current list of dictionaries include: force field parameter set names and types (e.g., classical, polarizable), enhanced sampling methods, MD analysis functions, barostats, thermostats, ensemble types, constraint algorithms, electrostatics models, basis sets and their types, calculation types (e.g., optimization, frequency, NMR), residues, atomic elements (periodic table) and their families, functional groups, software packages, and chemical file formats. The list also includes a dictionary of computational methods (e.g., Langevin dynamics, MP2, B3LYP) with their class (e.g., MD, Perturbation Theory, DFT) and family (e.g., ab initio, semiempirical, empirical). All these dictionaries are available for browsing and lookups at: http://ibiomes.chpc.utah.edu/dictionary/. Examples of dictionary entries are also provided in Appendix C.

Implementation

All our dictionaries follow the same implementation method. The raw data are defined in CSV files and can be loaded into a database for remote queries and/or indexed using Apache Lucene[20] for local access via Java APIs (Figure 3.12). Apache Lucene is a text search engine written in Java that uses high-performance indexing to enable exact and partial string matching. Each CSV file contains a list of entries for a given dictionary with at least three columns representing: the identifiers, the terms (e.g., “QM/MM”), and the term descriptions (e.g., “Hybrid computational method mixing quantum chemistry and molecular mechanics”). More columns can be defined depending on the type of dictionary, either to represent extra attributes or to link to other dictionaries (foreign keys). For example the CSV file listing the QM method classes would have an extra column with the IDs of the associated QM method families. A set of SQL scripts was written to automatically create the database schema necessary to store the dictionaries and to load the CSV data into the tables. These scripts become very useful if one wants to integrate these dictionaries into a repository. Another script was written to automatically build the Lucene indexes. The script calls a Java API which parses the CSV files and uses the Lucene API to build the indexes. These indexes can then be used locally by external codes via the Lucene API, avoiding the need for static definitions of these dictionaries within the code or the creation of dependencies with remote resources such as a database. They should also help future developments of chemical file parsers and text processing tools for chemical information extraction from the literature (i.e., natural language processing). The Lucene-based dictionaries can be directly queried through a simple command-line interface. Examples in Appendix D demonstrate how one would look up a term using this program. This design is fairly simple and enables updates of the dictionary entries directly through the CSV files. One limitation is the lack of synonyms for the terms defined. To create richer lists it will be necessary to add an extra CSV file for each dictionary that would contain the list of all the synonyms and the ID of the associated terms. Successful implementations of terminologies in other domains, such as the UMLS[42] (Unified Medical Language System), should be used to guide the organization of the raw data and facilitate the integration of existing terminologies representing particular aspects of the biomolecular simulations (e.g., chemical data, biomolecules, citations).

Maintenance and community support

Until this point the development of the dictionaries has been restricted to an internal effort by our lab. To support the work of the community at large these dictionaries have to be extended and adjusted based on user feedback. For this purpose the dictionaries are now available on our project Wiki at http://ibiomes.chpc.utah.edu/mediawiki/index.php/Dictionary, which enables discussions and edits by identified users. This will serve as a single endpoint to draft new versions of the dictionaries. The source code for the dictionaries, including the CSV files, SQL scripts, and Java API, is available from GitHub at: https://github.com/jcvthibault/biosim-repository. Updates on the CSV files hosted there should occur according to the status of the dictionaries in the Wiki. With time we might find that a dedicated database with a custom user interface becomes necessary for a defined group of editors to update existing terms, add new entries, add new dictionaries, and keep track of changes (logs). In any case, the number of editors should be limited to a small group of experts, actively participating and working together.[43, 44]

Discussion

In this paper we introduced a set of common data elements and a logical data model for biomolecular simulations. The model was built upon community needs, identified through a survey and refined internally. Elements described by the model cover the concepts of authorship, molecular system, computational method and platforms. Although the model presented here might not be complete, it integrates the methods that are the most significant for simulations of biomolecular systems: molecular dynamics, quantum chemistry and QM/MM. We introduced a new representation of the method landscape through method-specific parameter sets, which should allow the integration of more computational methods in the future. The addition of extended attributes to the model should enable customization by labs to fit their specific needs or represent properties that are currently not described by the model. The use cases presented here showed how the model can be used in real applications, to partially automate the creation of database schemas and generate XML descriptions. Multiple dictionaries, populated through reviews of online resources and literature, were implemented to supplement the model and provide developers with new tools to facilitate text extraction from chemical files and population of repositories. Although the current version of the dictionaries is fairly exhaustive they will become a powerful tool only if they are updated by the community. A missing piece in this model is a catalogue of available force field parameter sets and atom types that could be used to generate force field description files and serve as an input for popular MD software packages. The EMSL Basis Set Exchange[45] already offers something similar for basis sets, and provides a SOAP-based web service to access the data computationally.

While it is important to allow the whole community to provide input on the CDEs and dictionaries, eventually a consensus needs to be made by a group of experts representing the main stakeholders: simulation engine developers, data repository architects, and users. The creation of a consortium including users, developers and informaticians from the QM and the MD community could help formalize this process if such entity leads:

- Active polling, for example via annual surveys assessing the need for changes or additions in the CDEs, dictionaries, or the data model. Information about the respondents such as software usage, preferred computational methods (e.g., all-atom or coarse-grain MD, DFT) and target systems (e.g., chemical compounds, biomolecules) will provide more details for the development of more adequate recommendations for specialized communities.
- Monitoring of community discussions, which might take place on a dedicated online forum or a wiki such as the one introduced here
- Recurring creation and distribution of releases for the CDEs, dictionaries, and data model. The CDEs in particular should include at least 2 levels of importance (recommended or optional) to provide some criteria about the completeness of the data descriptors. A third level characterizing certain CDEs as mandatory might provide a standard for developers and data publishers to populate repositories.

Our current focus is on indexing data at the experiment level so that the associated collection of input and output files can be retrieved. While the CDEs can be used to tag individual files it is not clear yet how much metadata are necessary to enable automatic data extraction (e.g., extract properties for a single frame from a time series) and processing, and if such metadata can be extracted directly from the files without user input. The popularization of self-explanatory formats (e.g., NetCDF, CML) to store calculation results or MD trajectories would certainly help. The ongoing work within the ScalaLife programme should help the community move in this direction, while the data model presented here will provide a good framework to organize, describe, and index computational experiments comprising multiple tasks. By publishing this model and the list of CDEs we hope to encourage developments of new repositories for biomolecular simulations, whether they are part of an integrated computational environment (e.g., MDWeb) or not (e.g., iBIOMES). Both approaches should be addressed. On one hand, computational environments can easily keep track of the tasks performed during an experiment since the input parameters and topologies are directly specified within the environment. On the other hand, we still need to think about the developer community that works on new simulation engines, new force fields and new computational methods. They will still need to customize their simulation runs within more flexible environments where they can manually edit input files or compile new codes, and use local or allocated high-performance computing resources. Independent data repositories where data can be deposited through a publication process are probably more viable to overcome these requirements. Finally it is not clear who will be given access to these large computational environments or who will have the computational, storage, and human resources to deploy, sustain, and make such complex systems available to the community.

The goal of the proposed data model is to lay the foundations for a standard to represent biomolecular simulations, from the experiment level to the task level. For this purpose we wanted to integrate MD, QM, and QM/MM methods, all of which play a particular role in the field. Although classical MD is arguably the most popular approach for biomolecular simulations we believe that QM/MM approaches and ab initio MD for example will gain more and more interest as computational power increases and they should not be left out of a future standard. On the other hand we recognize that our model might not be as granular as others. The UMM XML[26] schema for example will be one of the first attempts to describe MD simulation input with enough granularity so that software-specific input files can be generated without information loss. Such effort is highly valuable for the MD community, and our data model will certainly evolve to integrate such models. Our short-term goal is to engage current repository and data model developers such as the ScalaLife (http://www.scalalife.eu/) and Mosaic (https://bitbucket.org/molsim/mosaic/wiki/Home) groups for MD and the Blue Obelisk (http://sourceforge.net/apps/mediawiki/blueobelisk/) group for QM and cheminformatics so that we can learn more about each other's experience and try to align our effort towards an integrated data model that would fit the needs of the whole biomolecular simulation community.

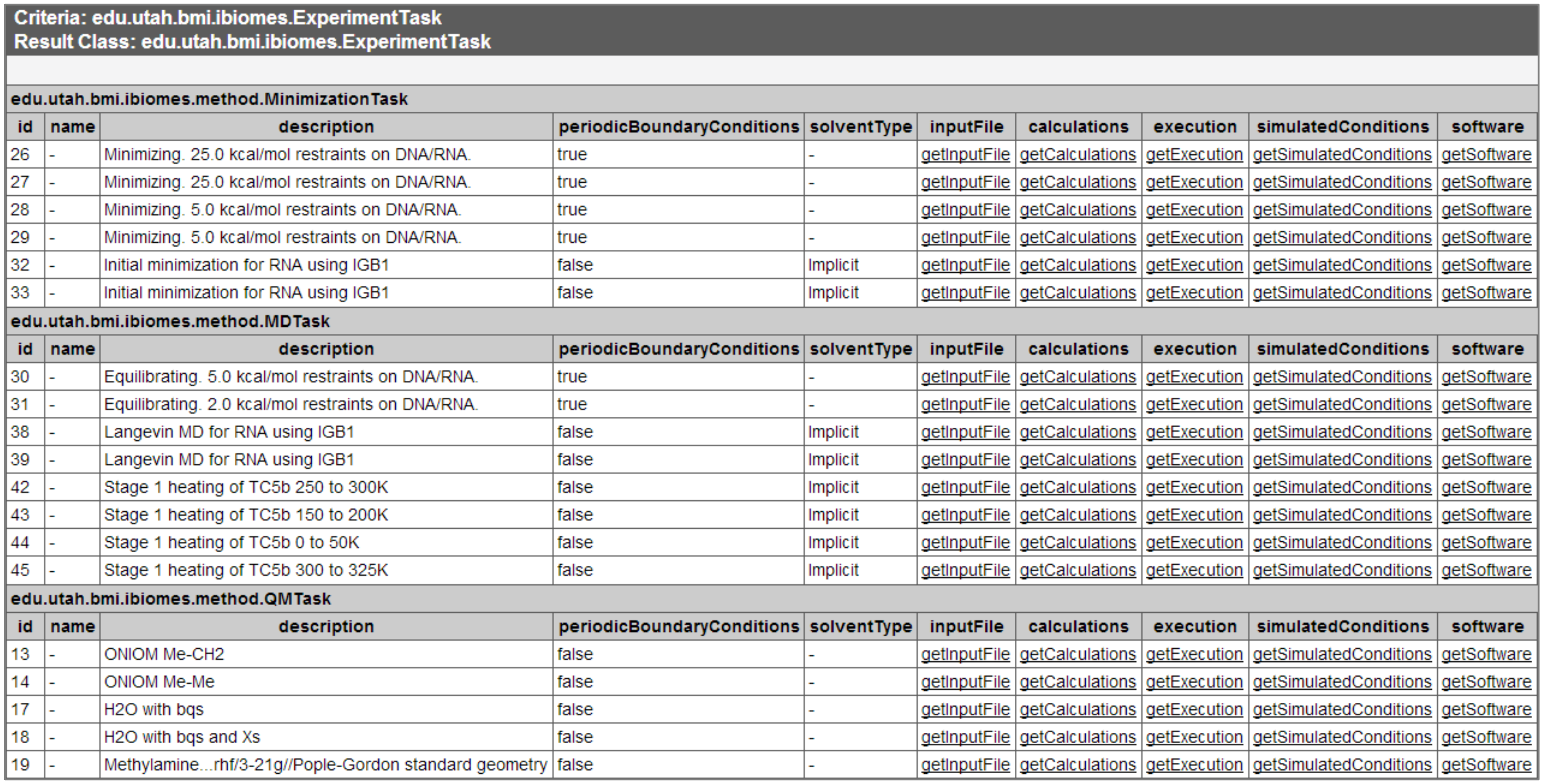
Conclusion

The framework presented here introduces a data model and a list of dictionaries built upon community feedback and selected experts' experience. The list of core data elements, the models, and the dictionaries are available on our wiki at: http://ibiomes.chpc.utah.edu/mediawiki/.

As more implementation efforts are taken, the community will be able to assess the present data model more accurately and provide valuable feedback to make it evolve, and eventually support collaborative research. The list of desiderata for data model developments, for both conceptual and physical representations, should provide some guidance for the long task at play.

Methods

This paper uses semistructured interview methods to establish the community needs and preferences regarding biomolecular simulation data indexing and presentation. The common data elements were identified using an approach similar to [46], while the data model was built using standard modelling techniques to derive logical and physical models. Interested readers can find details of these techniques in [22].



Criteria: edu.utah.bmi.ibiomes.ExperimentTask
Result Class: edu.utah.bmi.ibiomes.ExperimentTask

edu.utah.bmi.ibiomes.method.MinimizationTask

id	name	description	periodicBoundaryConditions	solventType	inputFile	calculations	execution	simulatedConditions	software
26	-	Minimizing. 25.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
27	-	Minimizing. 25.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
28	-	Minimizing. 5.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
29	-	Minimizing. 5.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
32	-	Initial minimization for RNA using IGB1	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
33	-	Initial minimization for RNA using IGB1	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware

edu.utah.bmi.ibiomes.method.MDTask

id	name	description	periodicBoundaryConditions	solventType	inputFile	calculations	execution	simulatedConditions	software
30	-	Equilibrating. 5.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
31	-	Equilibrating. 2.0 kcal/mol restraints on DNA/RNA.	true	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
38	-	Langevin MD for RNA using IGB1	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
39	-	Langevin MD for RNA using IGB1	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
42	-	Stage 1 heating of TC5b 250 to 300K	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
43	-	Stage 1 heating of TC5b 150 to 200K	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
44	-	Stage 1 heating of TC5b 0 to 50K	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
45	-	Stage 1 heating of TC5b 300 to 325K	false	Implicit	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware

edu.utah.bmi.ibiomes.method.QMTask

id	name	description	periodicBoundaryConditions	solventType	inputFile	calculations	execution	simulatedConditions	software
13	-	ONIOM Me-CH2	false	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
14	-	ONIOM Me-Me	false	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
17	-	H2O with bqs	false	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
18	-	H2O with bqs and Xs	false	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware
19	-	Methylamine...rhf/3-21g//Pople-Gordon standard geometry	false	-	getInputFile	getCalculations	getExecution	getSimulatedConditions	getSoftware

Figure 3.1, Screenshot of the web interface generated via the caGrid tools. The screenshot presents a listing of the computational tasks that were published into the caGrid test system. The user request was automatically translated into an SQL query via Hibernate to return the rows form the tables mapping to the class ExperimentTask and its child classes MinimizationTask (minimizations), MDTask (MD runs), and QMTask (QM calculations). For each row, a set of get methods (e.g., getSoftware) link to the associated objects for more details (e.g., Software name and version).

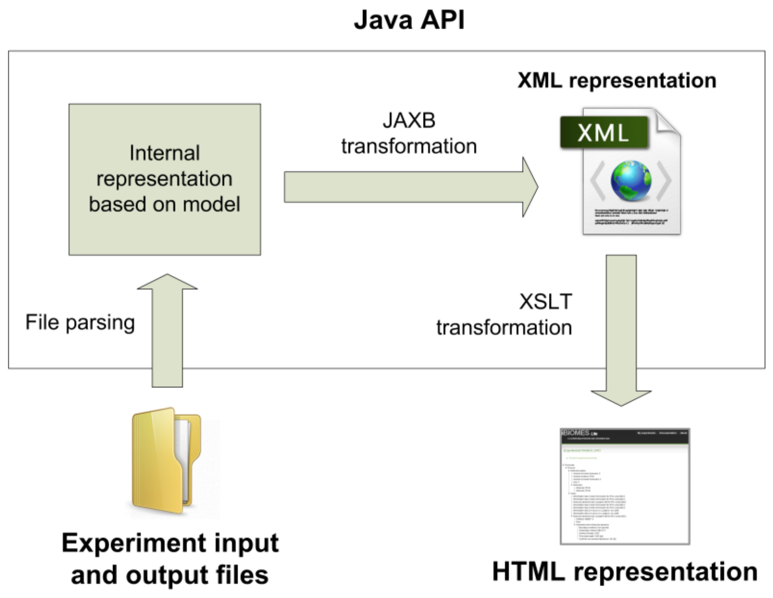
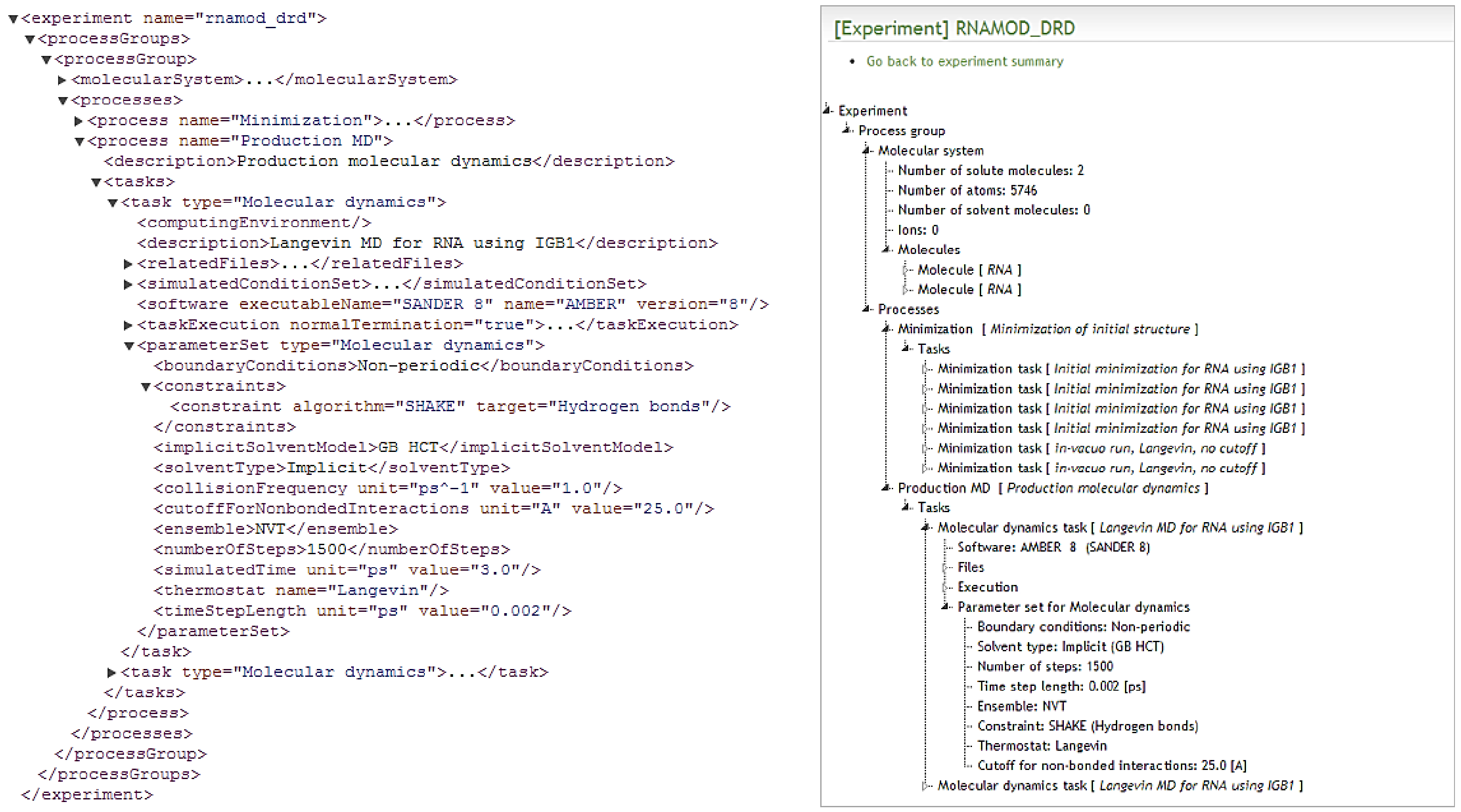


Figure 3.2, Generating an XML representation of experiments using a Java API. The Java API is used to parse the input files and create an internal representation of the virtual experiment as a set of computational tasks. JAXB is then used to generate an XML representation of this internal model, while XSLT is used to perform a last transformation into a user-friendly HTML page.



```
▼<experiment name="rnamod_drd">
 ▼<processGroups>
  ▼<processGroup>
   ▶<molecularSystem>...</molecularSystem>
   ▼<processes>
    ▶<process name="Minimization">...</process>
    ▼<process name="Production MD">
      <description>Production molecular dynamics</description>
     ▼<tasks>
      ▼<task type="Molecular dynamics">
         <computingEnvironment/>
         <description>Langevin MD for RNA using IGB1</description>
        ▶<relatedFiles>...</relatedFiles>
        ▶<simulatedConditionSet>...</simulatedConditionSet>
         <software executableName="SANDER 8" name="AMBER" version="8"/>
        ▶<taskExecution normalTermination="true">...</taskExecution>
        ▼<parameterSet type="Molecular dynamics">
           <boundaryConditions>Non-periodic</boundaryConditions>
          ▼<constraints>
             <constraint algorithm="SHAKE" target="Hydrogen bonds"/>
           </constraints>
           <implicitSolventModel>GB HCT</implicitSolventModel>
           <solventType>Implicit</solventType>
           <collisionFrequency unit="ps^-1" value="1.0"/>
           <cutoffForNonbondedInteractions unit="A" value="25.0"/>
           <ensemble>NVT</ensemble>
           <numberOfSteps>1500</numberOfSteps>
           <simulatedTime unit="ps" value="3.0"/>
           <thermostat name="Langevin"/>
           <timeStepLength unit="ps" value="0.002"/>
         </parameterSet>
       </task>
      ▶<task type="Molecular dynamics">...</task>
      </tasks>
    </process>
   </processes>
  </processGroup>
 </processGroups>
</experiment>
```

[Experiment] RNAMOD_DRD

- Go back to experiment summary

```
Experiment
  Process group
    Molecular system
      Number of solute molecules: 2
      Number of atoms: 5746
      Number of solvent molecules: 0
      Ions: 0
      Molecules
        Molecule [ RNA ]
        Molecule [ RNA ]
    Processes
      Minimization  [ Minimization of initial structure ]
        Tasks
          Minimization task [ Initial minimization for RNA using IGB1 ]
          Minimization task [ Initial minimization for RNA using IGB1 ]
          Minimization task [ Initial minimization for RNA using IGB1 ]
          Minimization task [ Initial minimization for RNA using IGB1 ]
          Minimization task [ in-vacuo run, Langevin, no cutoff ]
          Minimization task [ in-vacuo run, Langevin, no cutoff ]
      Production MD  [ Production molecular dynamics ]
        Tasks
          Molecular dynamics task [ Langevin MD for RNA using IGB1 ]
            Software: AMBER  8  (SANDER 8)
            Files
            Execution
            Parameter set for Molecular dynamics
              Boundary conditions: Non-periodic
              Solvent type: Implicit (GB HCT)
              Number of steps: 1500
              Time step length: 0.002 [ps]
              Ensemble: NVT
              Constraint: SHAKE (Hydrogen bonds)
              Thermostat: Langevin
              Cutoff for non-bonded interactions: 25.0 [A]
          Molecular dynamics task [ Langevin MD for RNA using IGB1 ]
```

Figure 3.3, XML and HTML-based representations of an experiment. Auto-generated XML sample (left) and corresponding HTML tree view (right) representing the different tasks run for an MD study of RNA using the AMBER software package.

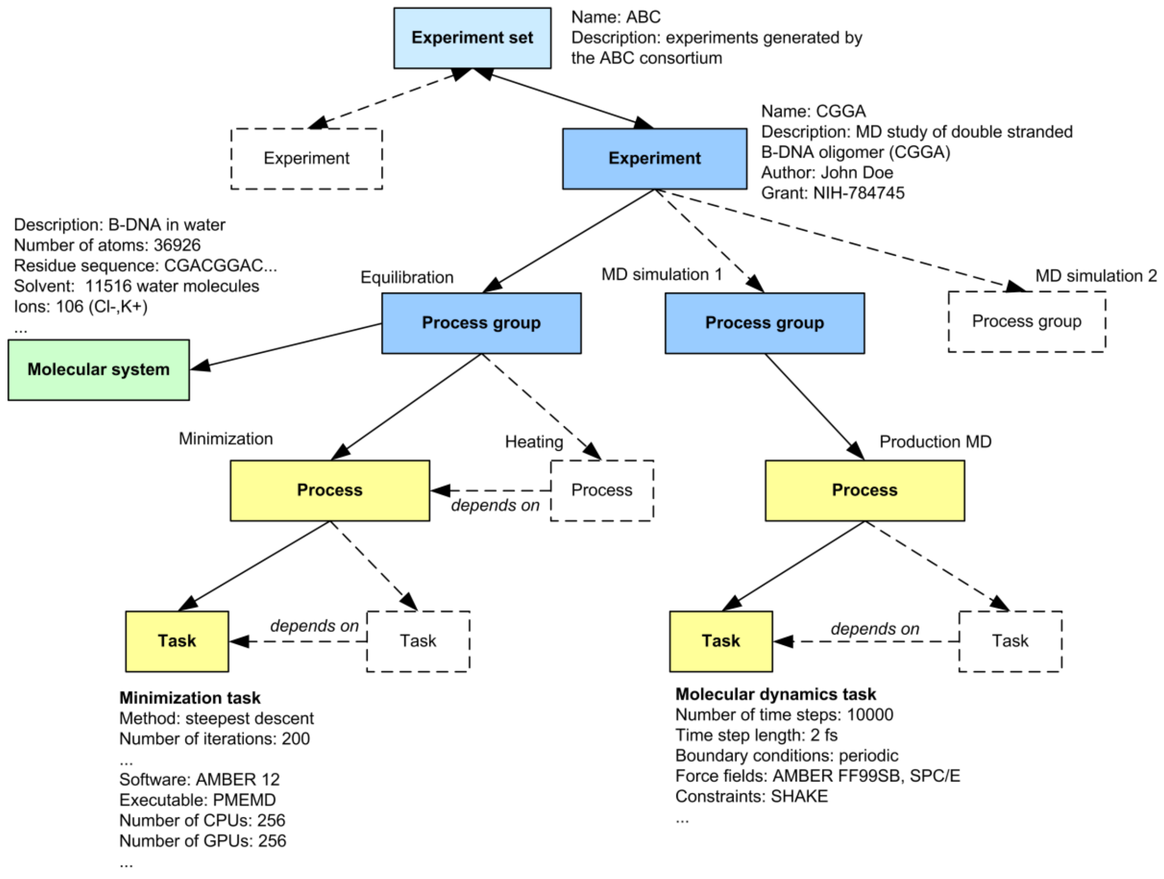


Figure 3.4, Illustration of the data model used to represent virtual experiments. Each experiment is a set of tasks, grouped into processes (e.g., minimization, equilibration, production MD) and process groups applied to the same molecular system (e.g., B-DNA oligomer).

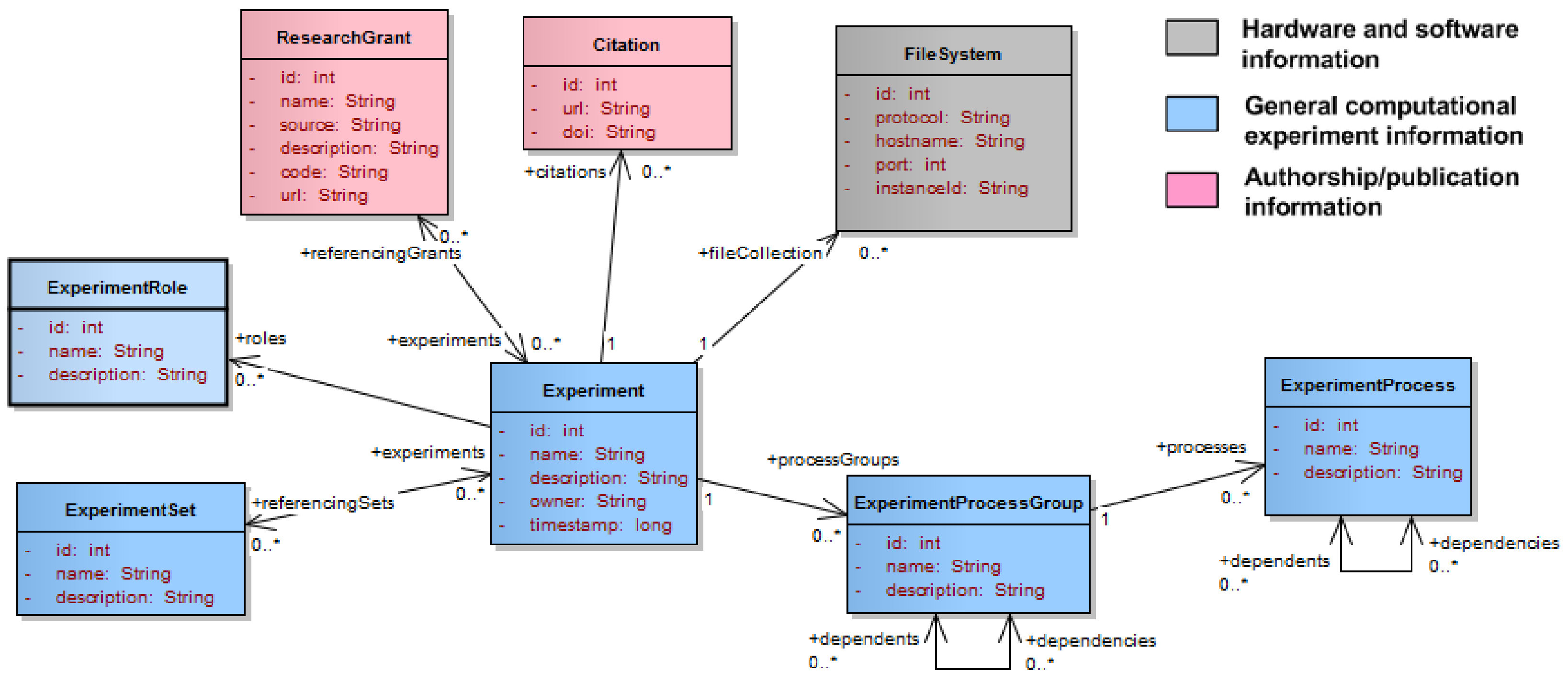


Figure 3.5, Concepts used to describe the context of the experiments.

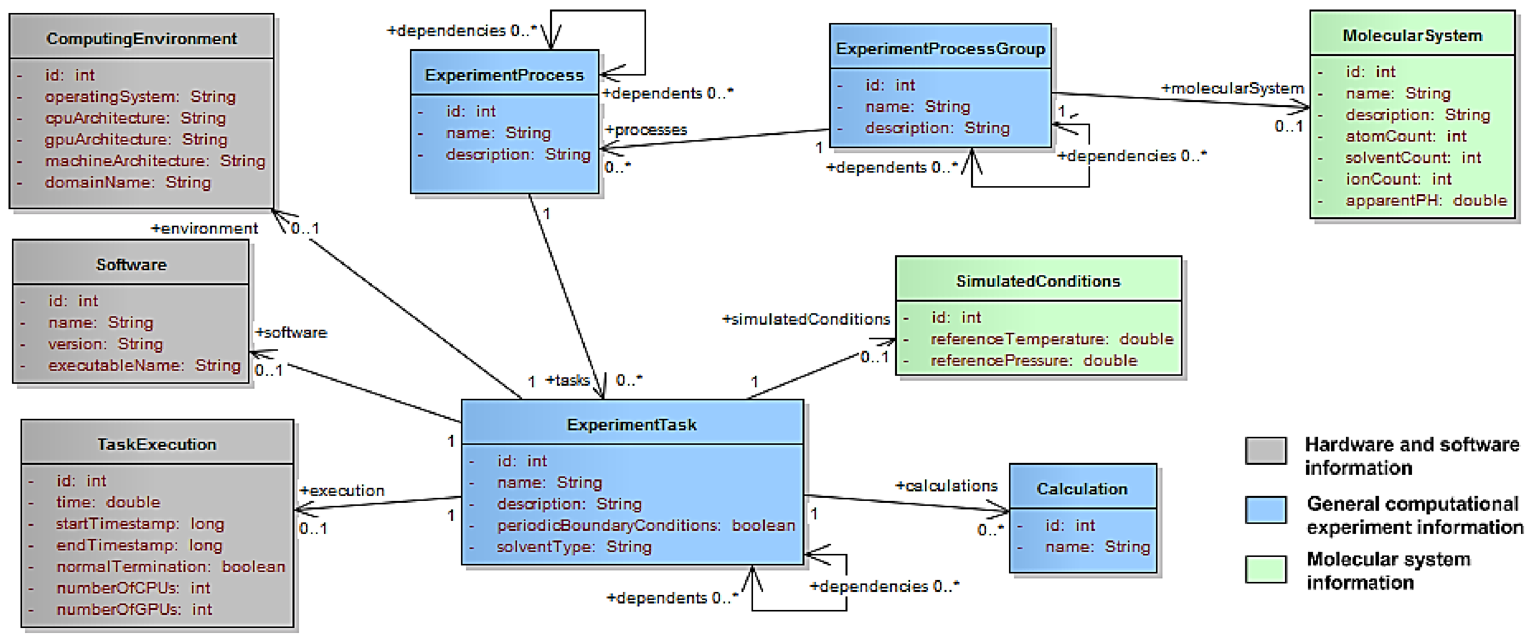


Figure 3.6, Description of experiments, processes, and tasks.

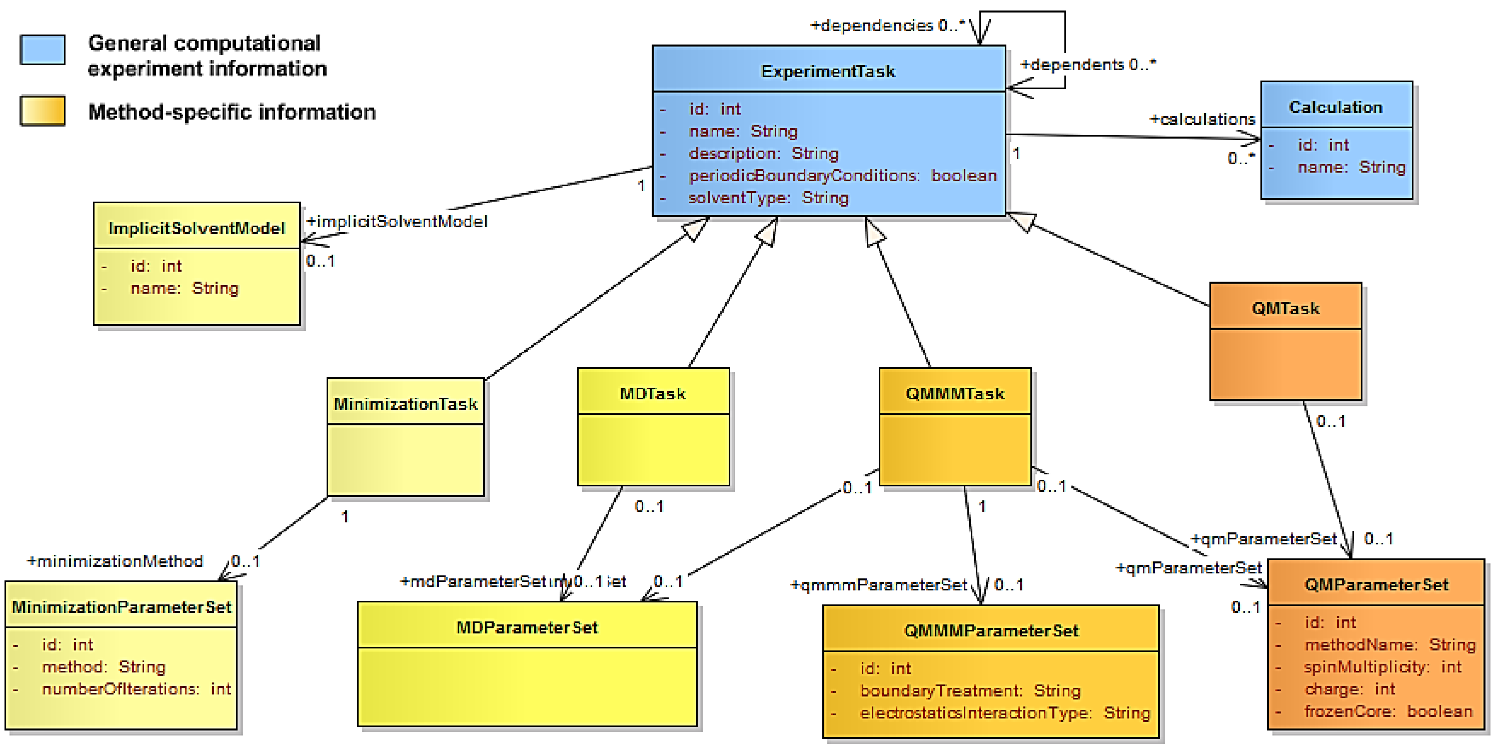


Figure 3.7, Organization of computational methods into tasks and parameter sets.

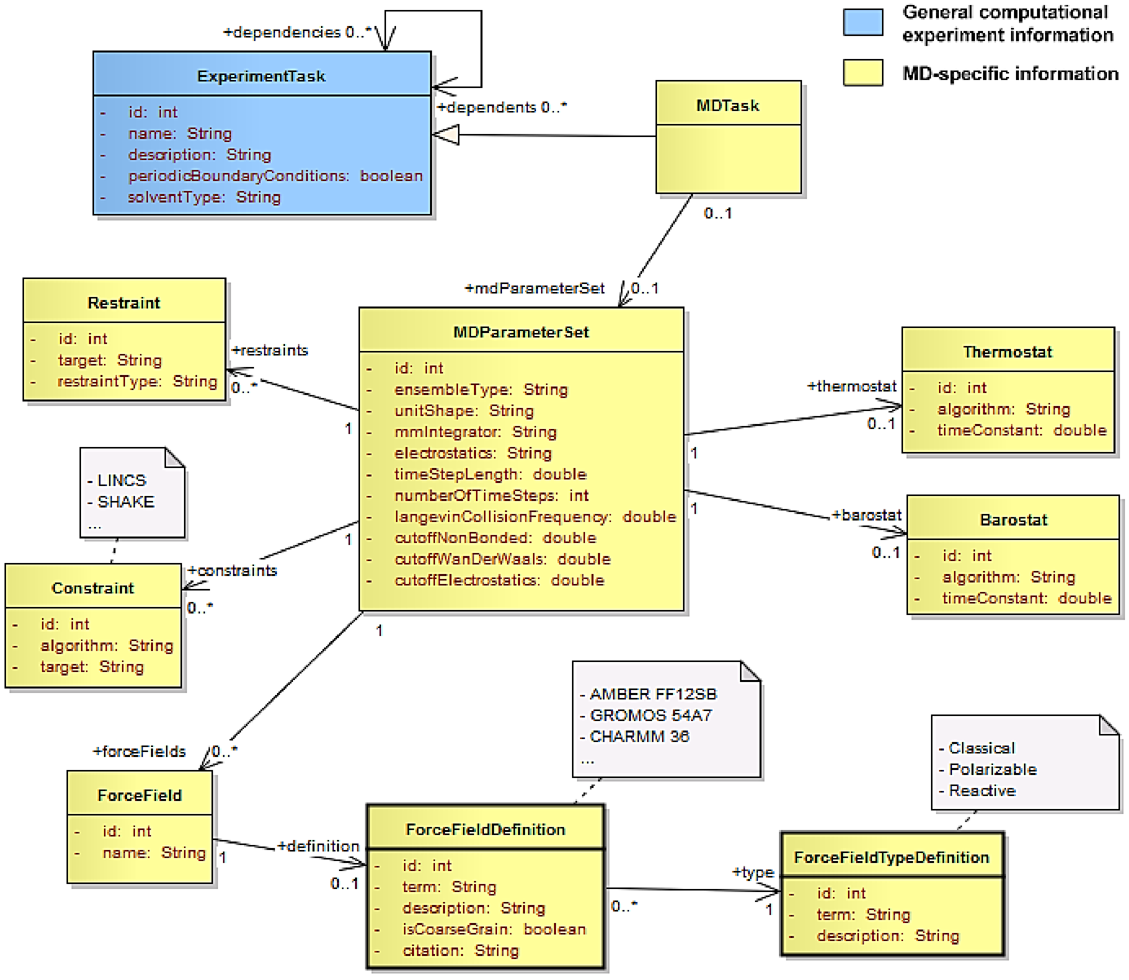


Figure 3.8, Description of MD tasks and parameter sets.

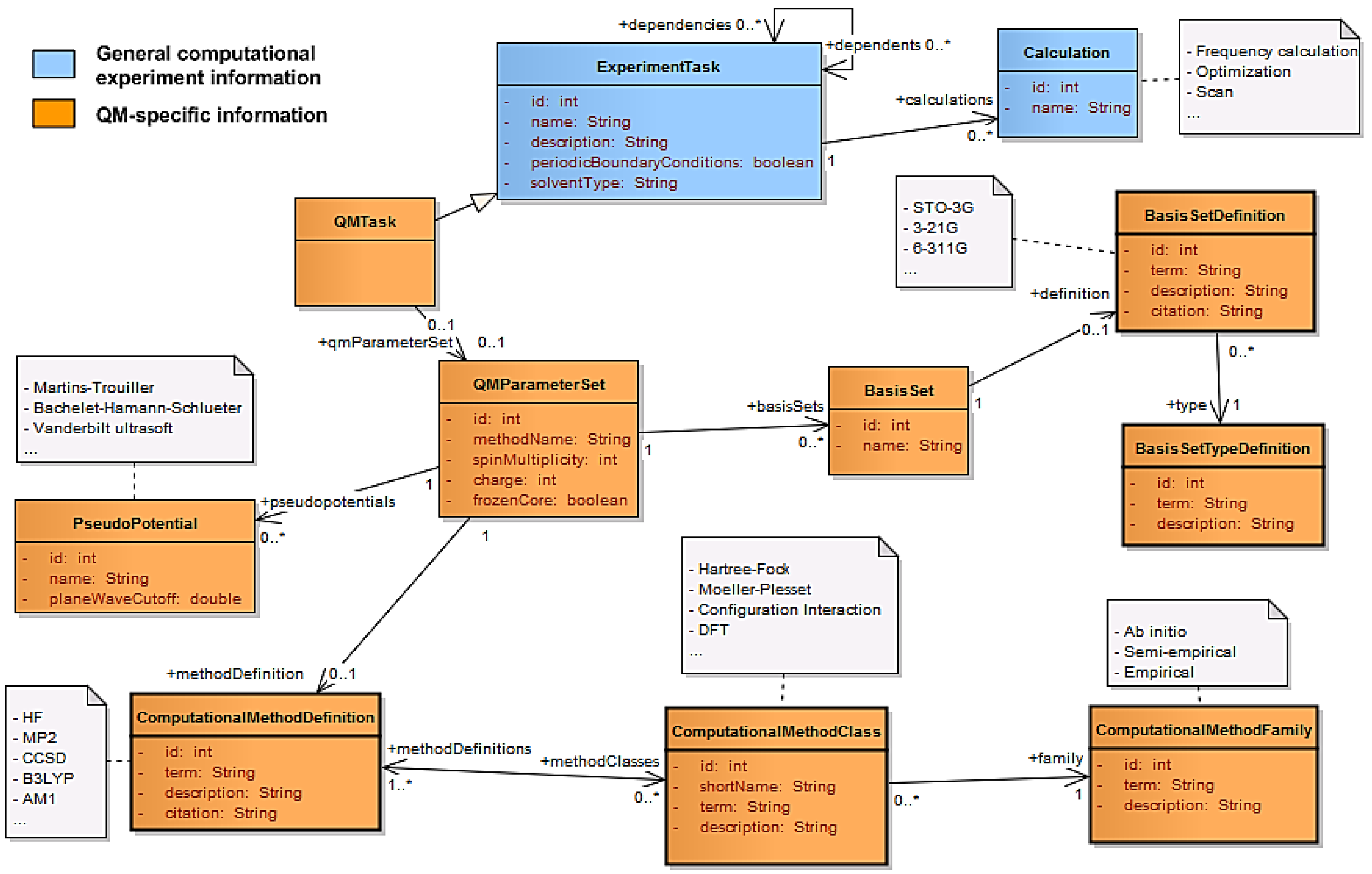


Figure 3.9, Description of QM tasks and parameters.

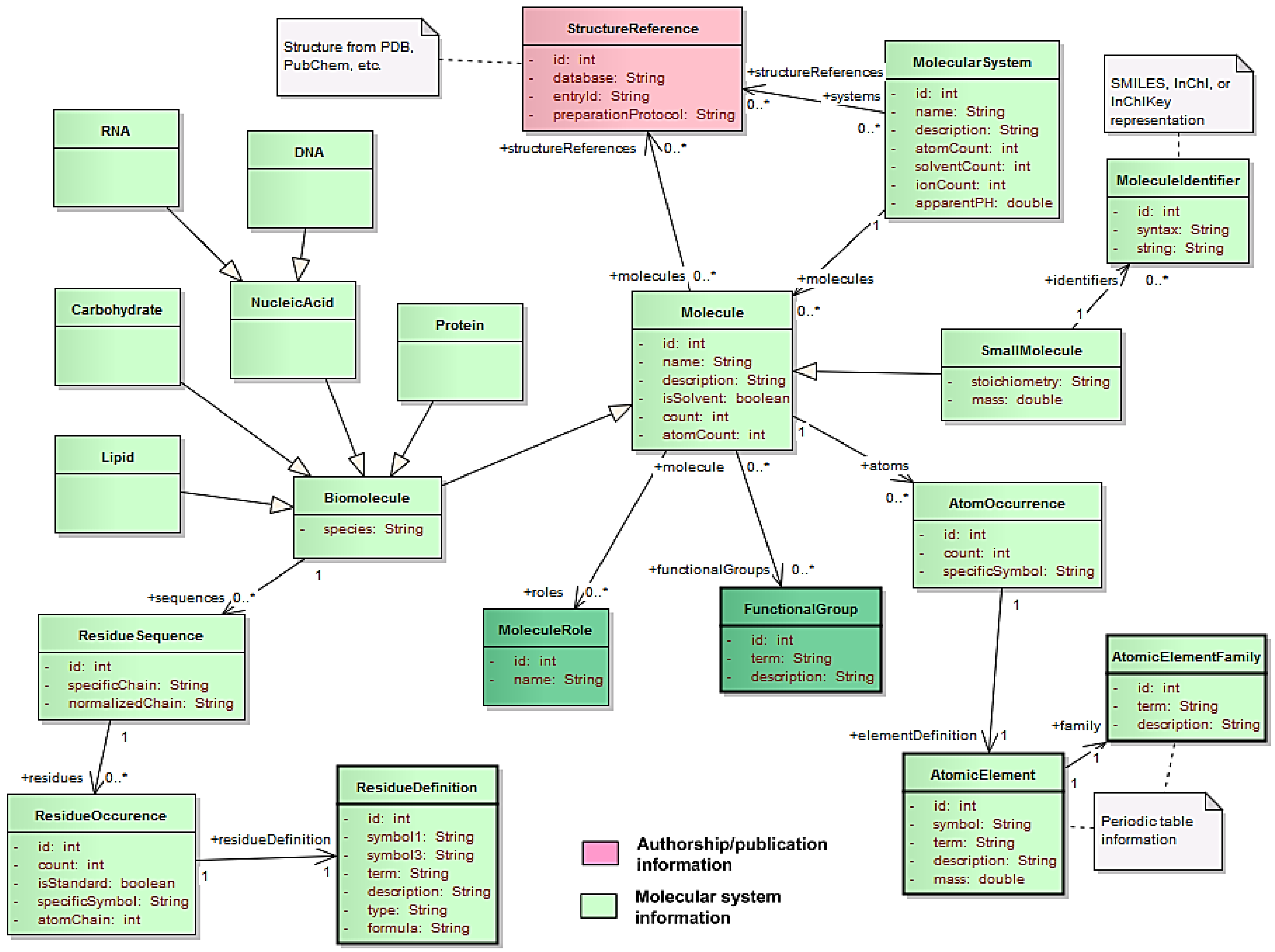


Figure 3.10, Decomposition of the molecular system into molecules with structural and biological features.

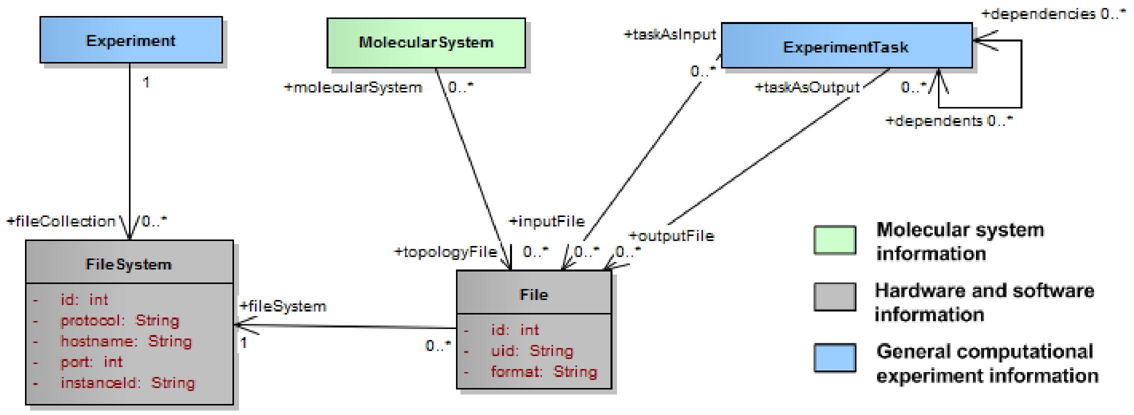


Figure 3.11, References to the file system and hosted files containing the raw data.

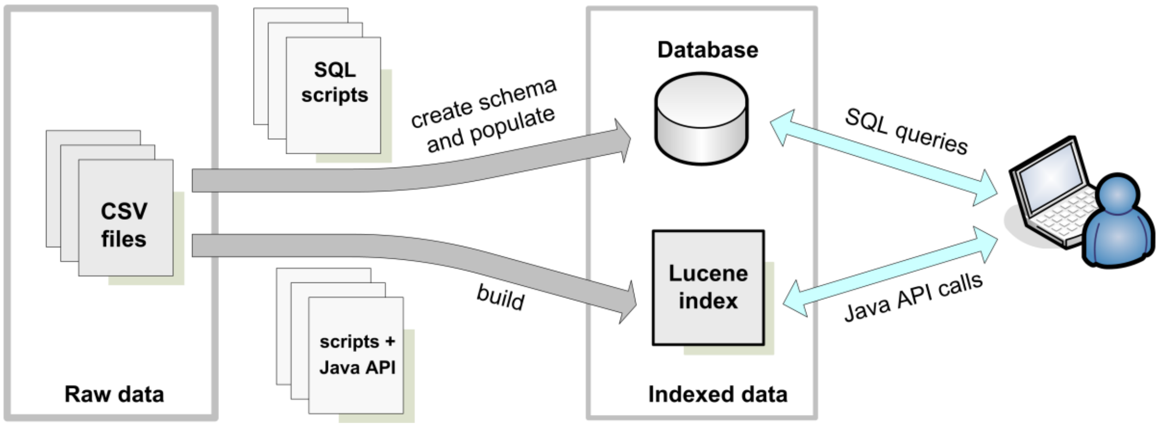


Figure 3.12, Building process for the dictionaries. Each dictionary can be either indexed via Apache Lucene for use via a Java API or loaded into a database to enable remote SQL queries.

References

1. Šponer, J.; Šponer, J. E.; Mládek, A.; Banáš, P.; Jurečka, P.; Otyepka, M., How to Understand Quantum Chemical Computations on DNA and RNA Systems? A Practical Guide for Non-Specialists. *Methods* **2013**, 64, 3-11.

2. Dror, R. O.; Dirks, R. M.; Grossman, J. P.; Xu, H.; Shaw, D. E., Biomolecular Simulation: a Computational Microscope for Molecular Biology. *Annu. Rev. Biophys.* **2012**, 41, 429-452.

3. Bernstein, F. C.; Koetzle, T. F.; Williams, G. J. B.; Meyer, E. F.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; Tasumi, M., The Protein Data Bank. *Eur. J. Biochem.* **2008**, 80, 319-324.

4. Simms, A. M.; Toofanny, R. D.; Kehl, C.; Benson, N. C.; Daggett, V., Dynameomics: Design of a Computational Lab Workflow and Scientific Data Repository for Protein Simulations. *Protein Eng. Des. Sel.* **2008**, 21, 369-377.

5. Toofanny, R. D.; Simms, A. M.; Beck, D. A.; Daggett, V., Implementation of 3D Spatial Indexing and Compression in a Large-Scale Molecular Dynamics Simulation Database for Rapid Atomic Contact Detection. *BMC Bioinformatics* **2011**, 12, 334.

6. Meyer, T.; D'Abramo, M.; Hospital, A.; Rueda, M.; Ferrer-Costa, C.; Perez, A.; Carrillo, O.; Camps, J.; Fenollosa, C.; Repchevsky, D.; Lluis Gelpi, J.; Orozco, M., MoDEL (Molecular Dynamics Extended Library): A Database of Atomistic Molecular Dynamics Trajectories. *Structure* **2010**, 18, 1399-1409.

7. Ng, M. H.; Johnston, S.; Wu, B.; Murdock, S. E.; Tai, K.; Fangohr, H.; Cox, S. J.; Essex, J. W.; Sansom, M. S. P.; Jeffreys, P., BioSimGrid: Grid-Enabled Biomolecular Simulation Data Storage and Analysis. *Future Gener. Comp. Sy.* **2006**, 22, 657-664.

8. Terstyanszky, G.; Kiss, T.; Kukla, T.; Lichtenberger, Z.; Winter, S.; Greenwell, P.; McEldowney, S.; Heindl, H., Application Repository and Science Gateway for Running Molecular Docking and Dynamics Simulations. *Stud. Health Technol. Inform.* **2012**, 175, 152-61.

9. Adams, S.; de Castro, P.; Echenique, P.; Estrada, J.; Hanwell, M. D.; Murray-Rust, P.; Sherwood, P.; Thomas, J.; Townsend, J., The Quixote Project: Collaborative and Open Quantum Chemistry Data Management in the Internet Age. *J. Cheminform.* **2011**, 3, 38.

10. Phadungsukanan, W.; Kraft, M.; Townsend, J. A.; Murray-Rust, P., The Semantics of Chemical Markup Language (CML) for Computational Chemistry : CompChem. *J. Cheminform.* **2012**, 4, 15.

11. Murray-Rust, P.; Rzepa, H. S., Chemical Markup, XML, and the World Wide Web. 4. CML Schema. *J. Chem. Inf. Comput. Sci.* **2003**, 43, 757-72.

12. Guha, R.; Howard, M. T.; Hutchison, G. R.; Murray-Rust, P.; Rzepa, H.; Steinbeck, C.; Wegner, J.; Willighagen, E. L., The Blue Obelisk-Interoperability in Chemical Informatics. *J. Chem. Inf. Model.* **2006**, 46, 991-8.

13. de Jong, W. A.; Walker, A. M.; Hanwell, M. D., From Data to Analysis: Linking NWChem and Avogadro with the Syntax and Semantics of Chemical Markup Language. *J. Cheminform.* **2013**, 5, 25.

14. Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. *Gaussian 09*, Revision C. 01; Gaussian, Inc: Wallingford, CT, 2009.

15. Valiev, M.; Bylaska, E. J.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Van Dam, H. J. J.; Wang, D.; Nieplocha, J.; Apra, E.; Windus, T. L., NWChem: A Comprehensive and Scalable Open-Source Solution for Large Scale Molecular Simulations. *Comput. Phys. Commun.* **2010**, 181, 1477-1489.

16. Thibault, J. C.; Facelli, J. C.; Cheatham, T. E., 3rd, iBIOMES: Managing and Sharing Biomolecular Simulation Data in a Distributed Environment. *J. Chem. Inf. Model.* **2013**, 53, 726-736.

17. Rajasekar, A.; Moore, R.; Hou, C.; Lee, C. A.; Marciano, R.; de Torcy, A.; Wan, M.; Schroeder, W.; Chen, S. Y.; Gilbert, L., iRODS Primer: Integrated Rule-Oriented Data System. *Synthesis Lectures on Information Concepts, Retrieval, and Services* **2010**, 2, 1-143.

18. Abouzied, A.; Bajda-Pawlikowski, K.; Huang, J.; Abadi, D. J.; Silberschatz, A. HadoopDB in Action: Building Real World Applications. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, Indianapolis, IN, USA, 2010; ACM: Indianapolis, IN, USA, 2010; pp 1111-1114.

19. Thusoo, A.; Sarma, J. S.; Jain, N.; Shao, Z.; Chakka, P.; Zhang, N.; Antony, S.; Liu, H.; Murthy, R. Hive-A Petabyte Scale Data Warehouse Using Hadoop. In Data Engineering (ICDE), 2010 IEEE 26th International Conference on, Long Beach, CA, USA, 2010; IEEE: Long Beach, CA, USA, 2010; pp 996-1005.

20. Apache Lucene. https://lucene.apache.org/

21. Herráez, A., Biomolecules in the Computer: Jmol to the Rescue. *Biochem. Mol. Biol. Educ.* **2006**, 34, 255-261.

22. Tillmann, G., *A Practical Guide to Logical Data Modeling*. McGraw-Hill: New York, 1993; p xiii, 248 p.

23. *The Grid 2: Blueprint for a New Computing Infrastructure*. second ed.; Morgan Kaufmann: San Francisco, CA, 2003.

24. Saltz, J.; Oster, S.; Hastings, S.; Langella, S.; Kurc, T.; Sanchez, W.; Kher, M.; Manisundaram, A.; Shanbhag, K.; Covitz, P., caGrid: Design and Implementation of the Core Architecture of the Cancer Biomedical Informatics Grid. *Bioinformatics* **2006**, 22, 1910-6.

25. Sun, Y.; McKeever, S., Converting Biomolecular Modelling Data Based on an XML Representation. *J. Integr. Bioinform.* **2008**, 5.

26. Goni, R.; Apostolov, R.; Lundborg, M.; Bernau, C.; Jamitzky, F.; Laure, E.; Lindhal, E.; Andrio, P.; Becerra, Y.; Orozco, M.; Lluis Gelpi, J., Standards for Data Handling. *ScalaLife White Paper* **2013**.

27. Case, D. A.; Cheatham, T. E., 3rd; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M., Jr.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J., The Amber Biomolecular Simulation Programs. *J. Comput. Chem.* **2005**, 26, 1668-1688.

28. Hess, B.; Kutzner, C.; van der Spoel, D.; Lindahl, E., GROMACS 4: Algorithms for Highly Efficient, Load-Balanced, and Scalable Molecular Simulation. *J. Chem. Theory. Comput.* **2008**, 4, 435-447.

29. Humphrey, W.; Dalke, A.; Schulten, K., VMD: Visual Molecular Dynamics. *J. Mol. Graphics* **1996**, 14, 33-38.

30. Roe, D. R.; Cheatham III, T. E., PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data. *J. Chem. Theory Comput.* **2013**.

31. Michaud-Agrawal, N.; Denning, E. J.; Woolf, T. B.; Beckstein, O., MDAnalysis: A Toolkit for the Analysis of Molecular Dynamics Simulations. *J. Comput. Chem.* **2011**, 32, 2319-2327.

32. Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Lluis Gelpi, J., MDWeb and MDMoby: An Integrated Web-Based Platform for Molecular Dynamics Simulations. *Bioinformatics* **2012**, 28, 1278-1279.

33. Svensson, M.; Humbel, S.; Froese, R. D.; Matsubara, T.; Sieber, S.; Morokuma, K., ONIOM: A Multilayered Integrated MO+ MM Method for Geometry Optimizations and Single Point Energy Predictions. A Test for Diels-Alder Reactions and Pt (P (t-Bu) 3) 2+ H2 Oxidative Addition. *J. Phys. Chem.* **1996**, 100, 19357-19363.

34. Jorgensen, W. L.; Tirado-Rives, J., Potential Energy Functions for Atomic-Level Simulations of Water and Organic and Biomolecular Systems. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, 102, 6665-6670.

35. Nadler, W.; Brunger, A. T.; Schulten, K.; Karplus, M., Molecular and Stochastic Dynamics of Proteins. *Proc. Natl. Acad. Sci. U. S. A.* **1987**, 84, 7933-7.

36. Schlick, T., Molecular Dynamics-Based Approaches for Enhanced Sampling of Long-Time, Large-Scale Conformational Changes in Biomolecules. *F1000 biology reports* **2009**, 1, 51.

37. Cramer, C. J., *Essentials of Computational Chemistry : Theories and Models*. 2nd ed.; Wiley: Chichester, West Sussex, England ; Hoboken, NJ, 2004; p xx, 596 p.

38. Weininger, D., SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **1988**, 28, 31-36.

39. McNaught, A., The IUPAC International Chemical Identifier. *Chemistry International* 2006, pp 12-14.

40. Altschul, S. F.; Gish, W.; Miller, W.; Myers, E. W.; Lipman, D. J., Basic Local Alignment Search Tool. *J. Mol. Biol.* **1990**, 215, 403-10.

41. Degtyarenko, K.; De Matos, P.; Ennis, M.; Hastings, J.; Zbinden, M.; McNaught, A.; Alcántara, R.; Darsow, M.; Guedj, M.; Ashburner, M., ChEBI: A Database and Ontology for Chemical Entities of Biological Interest. *Nucleic Acids Res.* **2008**, 36, D344.

42. Bodenreider, O., The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic Acids Res.* **2004**, 32, D267.

43. Hardiker, N.; Kim, T. Y.; Bartz, C. C.; Coenen, A.; Jansen, K. Collaborative Development and Maintenance of Health Terminologies. In AMIA Annual Symposium, Washington DC, 2013; American Medical Informatics Association: Washington DC, 2013; pp 572-577.

44. Noy, N. F.; Tudorache, T. Collaborative Ontology Development on the (Semantic) Web. In AAAI Spring Symposium: Symbiotic Relationships between Semantic Web and Knowledge Engineering, 2008; AAAI Press: 2008; pp 63-68.

45. Schuchardt, K. L.; Didier, B. T.; Elsethagen, T.; Sun, L.; Gurumoorthi, V.; Chase, J.; Li, J.; Windus, T. L., Basis Set Exchange: A Community Database for Computational Sciences. *J. Chem. Inf. Model.* **2007**, 47, 1045-52.

46. Kawamoto, K.; Del Fiol, G.; Strasberg, H. R.; Hulse, N.; Curtis, C.; Cimino, J. J.; Rocha, B. H.; Maviglia, S.; Fry, E.; Scherpbier, H. J.; Huser, V.; Redington, P. K.; Vawdrey, D. K.; Dufour, J. C.; Price, M.; Weber, J. H.; White, T.; Hughes, K. S.; McClay, J. C.; Wood, C.; Eckert, K.; Bolte, S.; Shields, D.; Tattam, P. R.; Scott, P.; Liu, Z.; McIntyre, A. K., Multi-National, Multi-Institutional Analysis of Clinical Decision Support Data Needs to Inform Development of the HL7 Virtual Medical Record Standard. *AMIA Annu. Symp. Proc.* **2010**, 2010, 377-81.

CHAPTER 4

THESAURUS AND ONTOLOGY DEVELOPMENTS FOR BIOMOLECULAR SIMULATION DATA EXCHANGE

Abstract

The field of biomolecular simulation is at the crossroads of chemistry, biology and computer science. As such, semantic description of the data and provenance metadata is critical to enable effective data sharing among these scientific communities. Until now the number of repositories for biomolecular simulation has been limited and no standard is followed to enable data interoperability and integration within the semantic web, greatly reducing the ability to exchange data with noncomputational scientists. In this paper we present a new thesaurus used to describe concepts related to the computational methods, parameters, and output commonly used in biomolecular simulations. We also demonstrate how to extend the thesaurus to a Simple Knowledge Organization System and an application ontology following the Open Biological and Biomedical Ontology Foundry principles.

Introduction

Biomolecular simulations aim to study the dynamics of biomolecules and biomolecular processes through computer simulations. While the computational methods mainly rely on approximations to physics and chemistry principles, the results aim to advance biology and medicine by providing new insights into molecular structure and function[1, 2] and are becoming critical for enabling drug discovery.[3, 4] In the last decade researchers in the field of molecular simulations have been able to reach timescales and system sizes that are biologically relevant.[1, 5] As computational power increases, these simulations become more common, and new tools are necessary to share these data with other scientific communities. At present simulation data are usually confined at the level of a lab or to a relatively small group of researchers participating in a multilab project. Very seldom the data are shared with the community at large or with the method developers. Few informatics architectures have been proposed to allow researchers to store and expose their data.[6-8] Even fewer repositories are openly available to the community to retrieve existing biomolecular simulation datasets[9, 10] Some of the main constraints for the development of such repositories are the amount of data created by each simulation (~GB-TB), the distributed nature of the storage resources, partly because of use of external computational resources, such as national clusters. But fundamentally the lack of the semantic context for simulation data precludes their use by researchers outside of the immediate circle of collaborators of the producing lab. The need for a common data model to store and exchange biomolecular simulation data has been demonstrated in various studies[11-13] but the current approaches are limited to usage within the biomolecular simulation community. Semantic description of biomolecular simulation data and provenance metadata is critical to enable data sharing with other communities, especially between the fields of experimental and computational chemistry. In this paper we report our advances in improving the semantic description of concepts related to the computational methods, parameters, and output of biomolecular simulations. Semantic description would not only allow federations of repositories based on different architectures, but it would also allow researchers from other scientific domains, such as experimental chemistry or biology, to gain more productive access to simulations data via the semantic web.

In previous work[11] we introduced a data model and a set of dictionaries to represent various concepts associated to the input parameters and output of biomolecular simulations. One of the limitations of these dictionaries is the lack of flexibility to represent hierarchies, especially when defining computational methods at different levels of granularity (e.g., “MP2” vs “Perturbation theory” vs. “Quantum chemistry”). This can be avoided by including “is a” relationships between concepts to create a detailed taxonomy. Such taxonomy can be enriched with associative relationships (e.g., “simulates,” “is executed on”) to give more meaning to the concepts through a thesaurus. Examples of such taxonomies include the various sources of the UMLS Metathesaurus[14] (e.g., NCI thesaurus, SNOMED-CT). Finally the thesaurus can be supplemented with implicit associations and definitions within an ontology to allow reasoning and infer relationships between concepts.

In this study we introduce a new controlled vocabulary for biomolecular simulations, BIOSIO (BIOmolecular SImulation Ontology), which can be used to describe published data using semantic web components. Our experience with iBIOMES[6] showed that complex queries cannot be built if the tags (i.e., metadata) associated to the published experiments do not have any semantic meaning. For example if a user is looking for all simulations that use molecular dynamics, one should expect the query engine to search for both classical and ab initio MD simulations. This type of inference assumes the existence of a controlled vocabulary representing hierarchical relationships between available tags.

The controlled vocabulary is defined as a thesaurus stored as a relational database based on the UMLS Metathesaurus model[14] to facilitate a future integration with other standard biomedical terminologies. The thesaurus was converted to a Knowledge Organization System (KOS) encoded as a Simple Knowledge Organization System[15] (SKOS), a W3C recommendation for the publication of controlled vocabularies within the semantic web. Finally, the thesaurus was extended to a simple ontology, to integrate concepts, relationships, and axioms of well-known biomedical ontologies.

Methods

Scope

The BIOSIO thesaurus and ontology aim to represent the following concepts:

- Theoretical chemistry methods, including quantum chemistry and molecular dynamics
- Analysis methods (e.g., Root mean square deviation calculations, principal component analysis)
- Computational tasks, including input and output description
- Software packages and file formats

Theoretical and computational methods are not actually described by the ontology. Instead BIOSIO provides a reference to the associated literature or web content when applicable. Such references are also used when describing software packages and file formats. BIOSIO was implemented in 3 different formats: as a relational database, as a Simple Knowledge Organization System[15] (SKOS), and as an OWL 2 document.[16]

Implementation

Thesaurus database

A database was first designed to store the concepts represented by the data model and dictionaries presented in previous work and validated by the user community as explained elsewhere.[11] The database schema (Figure 4.1) was inspired by the UMLS metathesaurus.[14] Each concept (i.e., meaning) is defined in the CONCEPT table and can be associated to several terms (i.e., synonyms), citations, and textual descriptions. Some of the classes and attributes from the initial data model were used to manually create new concepts in the database. A set of scripts was created to automatically create a new concept with its textual description and citations (if applicable) for each dictionary entry. For example, a “computational method” concept was created to be the parent of “Ab initio methods,” “Empirical method,” and “Semiempirical method,” which were defined as part of the dictionary of computational methods. These concepts were supplemented with various concepts that did not appear in the original data model but that were necessary to bring more granularity to the hierarchical organization of the controlled vocabulary. For example, the force field parameter sets were grouped by publisher (e.g., AMBER, CHARMM) and targets (e.g., ions, water). Relationships between concepts are defined in the RELATIONSHIP table, while types of relationships (e.g., "is a," "has part") are defined in the CONCEPT table and differentiated from regular concepts via the IS_REL flag. Concepts can also be mapped to concepts from external terminologies or ontologies via the CONCEPT_MAPPING and EXTERNAL_ONTOLOGY tables, which store the mappings and ontology definitions respectively. The SEMANTIC_TYPE table stores the various categories used to provide a high-level classification of all concepts in the thesaurus: the semantic types. Each concept can be associated to one or multiple semantic types via the CONCEPT_SEMANTIC_TYPE table. Just like in the UMLS, semantic types are defined to reduce the complexity of the thesaurus.[17] They can be used to group similar concepts together and facilitate searches and result filtering. For the design of this thesaurus we created a simple semantic network that would enable targeted searches based on the different parameters and methods (i.e., molecular dynamics vs. quantum chemistry) one could choose to setup the simulation. Each concept in the thesaurus can be assigned to at least one semantic type.

SKOS and ontology

A Java API was developed to enable the creation of SKOS and OWL documents from the thesaurus defined in the relational database. The API queries the database and iterate through all the concepts to write the associated triples into a SKOS or OWL Turtle file.[18] The API can also be used to populate the database from a SKOS or OWL document, using the OWL API[19] and the SKOS API.[20] The following assumptions were made when developing the API: (1) High-level relationships such as "is a," and "has parts" are mapped using external ontologies (Table 4.1) that are assumed to be referenced in the thesaurus or the OWL ontology; (2) In SKOS, hierarchical relationships are represented through the “narrower” and “broader” associations. For example “DNA” is a “broader” concept than “Nucleic acid,” and “all-atom molecular dynamics” is narrower than “molecular dynamics.” In OWL, “is a” relationships are expressed using the subClass predicate. For example the “Nucleic acid” class is a sublcass of “DNA.”

The BIOSIO ontology development follows the principles of the Open Biological and Biomedical Ontology (OBO) Foundry, a group of developers aiming at creating interoperable ontologies for the biomedical domain. BIOSIO builds upon the Basic Formal Ontology[21] (BFO) as its upper-level ontology. BFO defines abstract concepts such as “continuant,” i.e., an entity that exists and persists through time (e.g., a material entity, a spatial region), and “occurent”, i.e., an entity that has temporal parts (e.g., a process, an event, a temporal region). These concepts serve as a foundation for most OBO ontologies to facilitate interoperability and future developments. BIOSIO also builds upon more concrete ontologies derived from BFO: the Information Artifact Ontology (IAO), which describes information entities such as data sets, documents, software and algorithms, and the Ontology for Biomedical Investigations (OBI), which aims to describe the wide spectrum of biological and clinical investigations, from their design to the analysis methods and resulting data sets.[22] BIOSIO, like many other OBO ontologies, uses the ChEBI[23] (Chemical Entity of Biological Interest) ontology to define chemical and molecular entities, such as atoms, ions, molecules, nanostructures, nucleic acids, and proteins. Biological concepts can be derived from ChEBI by linking to other OBO ontologies, such as the Gene Ontology[24] (GO) or the Protein Ontology[25] (PRO).

The final OWL document only stores references to these ontologies. One can explicitly import these ontologies via tools such as Protégé[26] if the associated concepts are necessary for the use case. The SKOS-encoded controlled vocabulary on the other hand does not include references to external sources, such as ChEBI, which is necessary to represent concepts related to molecular and chemical entities. Conversion tools such as *skosify* (https://code.google.com/p/skosify/) and the OBO-to-SKOS converter from the University of Manchester (http://www.cs.man.ac.uk/~sjupp/skos/index.html) could be used to generate a SKOS version of ChEBI and represent these missing pieces.

Comparison with the UMLS

One of the long-term goals for this thesaurus is to become part of a larger source of biomedical concepts such as the UMLS to supplement existing concepts with new concepts relating to biomolecular simulations. In order to evaluate the novelty of the concepts introduced in this thesaurus we compared the overlap between the UMLS concepts and the BIOSIO thesaurus concepts. A quantitative evaluation of this overlap was performed by looking at the matches between concept terms. If all concepts introduced in this thesaurus are novel no overlap should be found with the UMLS. On the other hand, matches help identify where mapping is necessary. To facilitate this process we developed a simple dictionary lookup program to automatically compare strings of concept names from both sources. About 10 million concept terms from the 2012AB UMLS were indexed using Apache Lucene,[27] a high-performance text search engine. A Java program based on the Apache Lucene API was developed to check exact matches between normalized concept terms from our thesaurus and the UMLS. The normalized version of a term is obtained by removing common stop words (e.g., “a”, “and”, “with” “to”) and by using the canonical form of each word using the Lexical Variant Generator[28] (LVG) tool. For example, plural nouns become singular, and conjugated verbs are transformed to their infinitive root. This normalization step is performed on each UMLS term when building the index and on each thesaurus concept term that is looked up in the index. This process tends to reduce the number of false negatives when comparing strings. To facilitate the analysis of the matches proposed by our program, each concept term in the index is associated to its CUI (Concept Unique Identifier), its original term, a normalized version of the term, and the source terminology for the concept (e.g., ICD-10, MESH, NCI).

SKOS use case

iBIOMES builds upon the iRODS[29] framework, which provides a distributed file system where files are indexed using Attribute-Value-Unit (AVU) triplets. One of the current directions undertaken by the iRODS developers is the integration of KOS within their indexing system. More specifically, they are in the process of integrating HIVE (Helping Interdisciplinary Vocabulary Engineering[30]) to manage and index SKOS-encoded controlled vocabularies. HIVE provides a core server to load SKOS documents and to enable keyword and SPARQL[31] searches. HIVE also supports automatic document tagging using keyphrase extraction, based on the KEA (Keyphrase Extraction Algorithm[32]) tool. Assuming that a model is trained within KEA, this could enable automatic biomolecular simulation literature tagging and indexing. To assess such framework within iBIOMES, we installed a local instance of HIVE and loaded the BIOSIO SKOS to enable concept browsing and searches.

Results

Concept network

Summary

In total 697 concepts (i.e., OWL classes) and 870 associated terms (i.e., OWL labels) are represented in BIOSIO. Twelve high-level concepts were mapped to external OBO ontologies, as listed in Table 4.2. For example the “software package” concept does not have any explicit parent in BIOSIO but it is mapped as a child of the concept “software” in the IAO ontology. All these parent-child mappings provide a higher level of abstraction for BIOSIO if integration with other biomedical ontologies is necessary.

The core concepts (i.e., without external ontology mappings) are organized through 677 “is a” relationships and 13 “has part” relationships. The resulting hierarchical network of core concepts is presented in Figure 4.2. Each node represents a concept explicitly defined in the thesaurus and each edge represent an “is a” relationship.

BIOSIO also includes 139 citations (127 unique references), most of which were already published in our dictionaries.[11] The thesaurus also includes 12 semantic types to provide a high-level classification of the concepts similar to the UMLS semantic type network. These semantic types were organized into a simple network, as illustrated in Figure 4.3. Each concept in BIOSIO is considered a simulation feature that relates to the computational methods (e.g., molecular dynamics and associated parameters), the molecular system (e.g., topology, structure) or the computing environment (i.e., software or hardware used to run the simulation).

Comparison with the UMLS

Out of the 697 BIOSIO thesaurus concepts, 94 had at least one term name that matched a UMLS metathesaurus concept name. Some of these term matches, including true and false positives, are presented in Table 4.3. Out of the 94 BIOSIO concepts being mapped by the program, 33 concepts were mapped correctly to either an equivalent or a parent UMLS concept. Most of the false positives were caused by acronyms that did not have the same meaning in both sources. For example the acronym SAS (surface-accessible surface) in BIOSIO matched different gene names ("NANS," "TSPAN31") that use this string as alternate identifiers in the UMLS. Most of the true positives are related to software or hardware components (e.g., CPU, GPU, file). This is expected since our thesaurus includes concepts related to the computing environment, but leaves out the description of biomolecular systems, which would have great overlap with the UMLS. Another source of false positives is the difference in granularity between matching concepts. For example the concept "Analysis task" in our thesaurus really represents computational analysis tasks, and not a generic "analysis" (C0936012) or "analysis of substances" (C0002778). Although we considered these mappings as false positives they can actually help identify child-parent mappings.

Indexing SKOS concepts with HIVE

The SKOS document was successfully validated using the online quality checker available at http://qskos.poolparty.biz/ and loaded into HIVE. A screenshot of the web interface of our local HIVE instance is presented in Figure 4.4. Although the original version of the SKOS successfully passed the quality tests, it did not fulfill all the requirements of the HIVE system to be successfully loaded. The database-to-SKOS converter had to be updated to 1) explicitly define each "is a" relationship with both "narrower" and "broader" associations (although in SKOS "A narrower B" implicitly means "B broader A"), 2) define a SKOS scheme (skos:ConceptScheme) for all the concepts (skos:inScheme) and explicitly define the top-level concepts (skos:hasTopConcept), and 3) define document-level metadata (e.g., creation date, author). An extract of the final document is given in Figure 4.5.

A few SPARQL[31] queries were run against HIVE using the HIVE-core Java API (version 2.2). Two example input SPARQL queries are provided in Figure 4.6 and Figure 4.7 to show how one would retrieve broader and narrower concepts.

Discussion

In this paper we presented a new controlled vocabulary for biomolecular simulations, BIOSIO, that focuses on the representation of the computational methods, parameters and environments (i.e., software and hardware) relating to biomolecular simulations. A preliminary analysis was performed to check for overlaps between this thesaurus and the UMLS, one of the largest sources of biomedical concepts. Our results show that a future integration of the BIOSIO thesaurus into the UMLS metathesaurus will require some manual work but semiautomatic mappings between concepts will facilitate the process. The precision of our current mapping algorithm, based on a simple index lookup, could be largely improved. For example one could remove acronyms from the automatic mapping step and rely only on expanded labels to compare strings. In our analysis we used the whole UMLS, although we are only interested in computational methods and computing environment components. To avoid false positives such as gene and protein names, we could filter out certain UMLS semantic types representing biomolecular and chemical entities, since these are not directly represented in our thesaurus and are not expected to match any concept. The recall of the algorithm will be highly dependent on the richness of the vocabularies being mapped. Even though the normalization step used for indexing and lookups should provide a good recall, some concept mappings might have been missed because of poor representation of synonyms for the associated concepts in either source.

A SKOS-encoded controlled vocabulary and a simple ontology were derived from the thesaurus. The SKOS was validated and loaded into HIVE to enable concept browsing and searches. Sample SPARQL queries were run to show the value of SKOS to expose biomolecular simulation data in a semantic web context. The derived ontology links to popular OBO ontologies to integrate detailed descriptions of biomolecule and chemical entities, but also for the integration of more abstract concepts that should facilitate its reuse in future OBO developments. Future directions include the integration of the ontology into the Chemical Information Ontology[33] (CHEMINF), which describes a domain that is similar to biomolecular simulations in many aspects. It aims to provide a description of cheminformatics tools and calculations within a semantic web context. This includes the description of the algorithms, their execution process, the input and output, and the actual chemical descriptors being calculated. Although CHEMINF focuses on cheminformatics applications, its higher-level concepts are adapted to most subfields of computational chemistry, including quantum chemistry and molecular dynamics, two of the main classes of methods for biomolecular simulations. Finally the current ontology presented here is very simple since it does not include any axiom other than the ones inherited from the parent ontologies (e.g., BFO, OBI). The inclusion of more associative relationships and axioms specific to the domain of biomolecular simulation should help infer certain characteristics of computational experiments. For example, when publishing incomplete metadata into a repository, a reasoner such as HermiT (http://hermit-reasoner.com/) or Pellet (http://clarkparsia.com/pellet/) could be used to generate missing or more specific metadata.

At this point the concepts and relationships defined in the thesaurus and the ontology have not been formally evaluated, although they build upon a previously published data model.[11] This work was mostly done within the context of a single computational lab and did not involve outside experts. A survey could be used to receive general feedback but a detailed evaluation using a divide-and-conquer approach might be more beneficial. For example a group of experts would be responsible to evaluate and refine the ab initio methods while another group would be responsible for the classical MD methods. More complete methodologies for the evaluation of controlled vocabularies and ontologies could be used.[34, 35] Coverage of the domain should be evaluated as well. There are numerous computational methods and parameters one can use to run biomolecular simulations. The computational protocols are rarely described in detail in the literature, which usually prevents reproducibility. Automatic term extraction using existing algorithms[36] could be useful to generate a list of common terms that represent biomolecular simulation methods. Since the associated literature is usually focusing on a higher level of theory and on the actual results of the simulations, the use of various QM and MD software user manuals might be more adapted to the scope of our work.

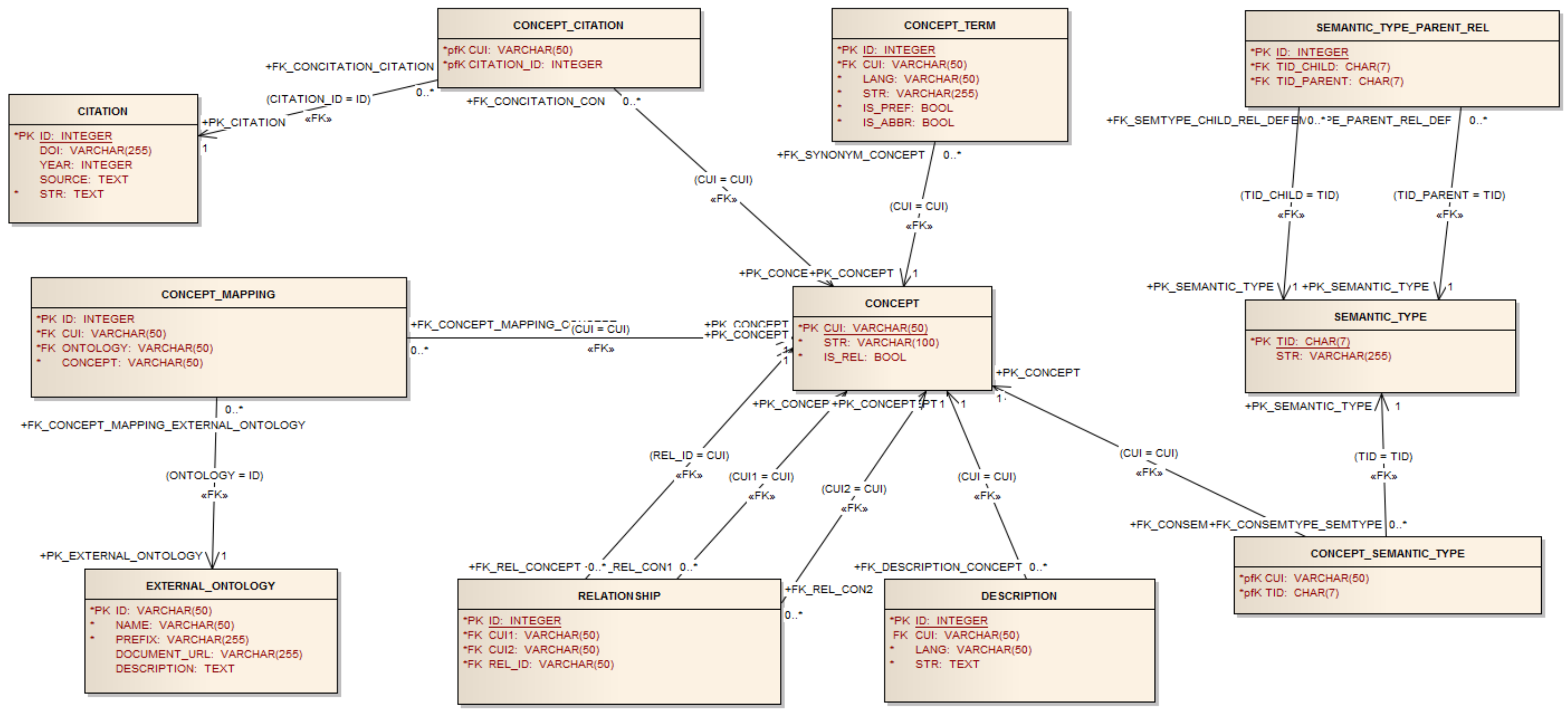


Figure 4.1, Database schema for the thesaurus

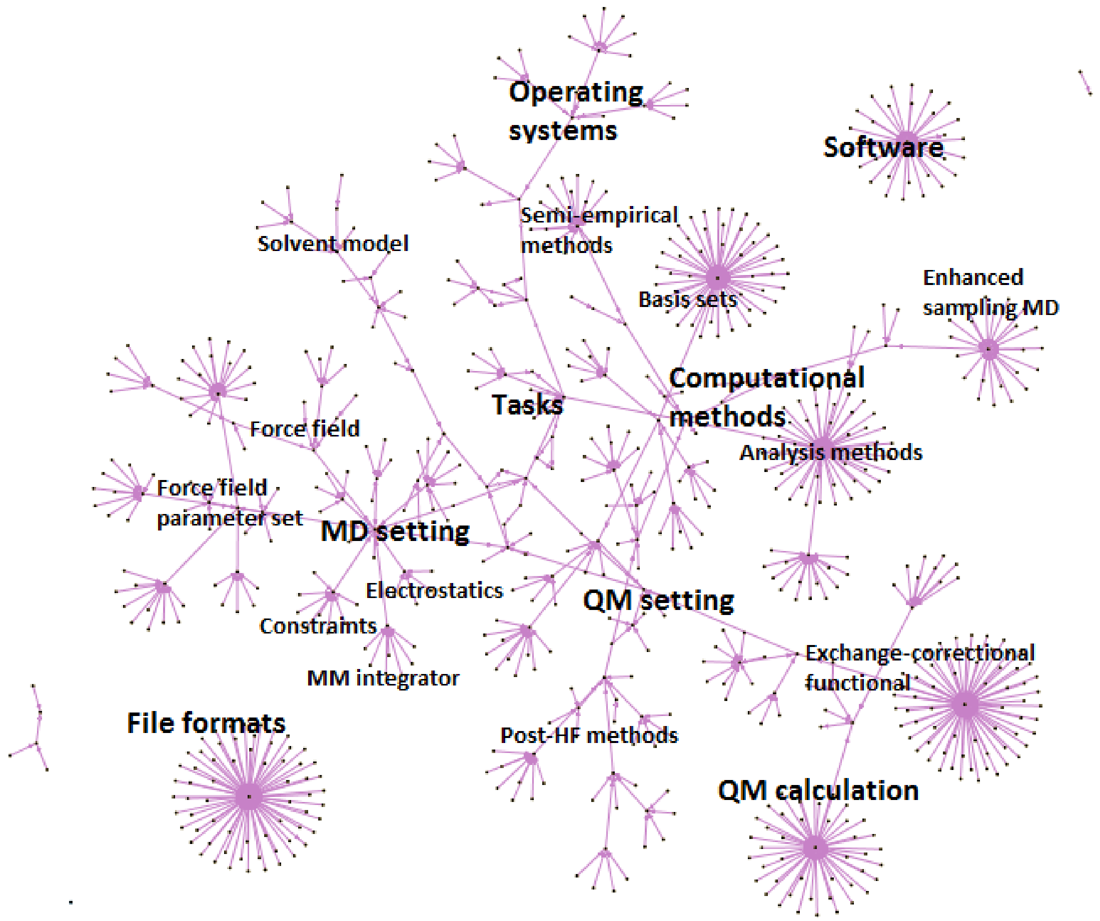


Figure 4.2, Hierarchical network of BIOSIO core concepts without external ontology mappings. Each leaf represents a concept and each branch represents an “is a” relationship.

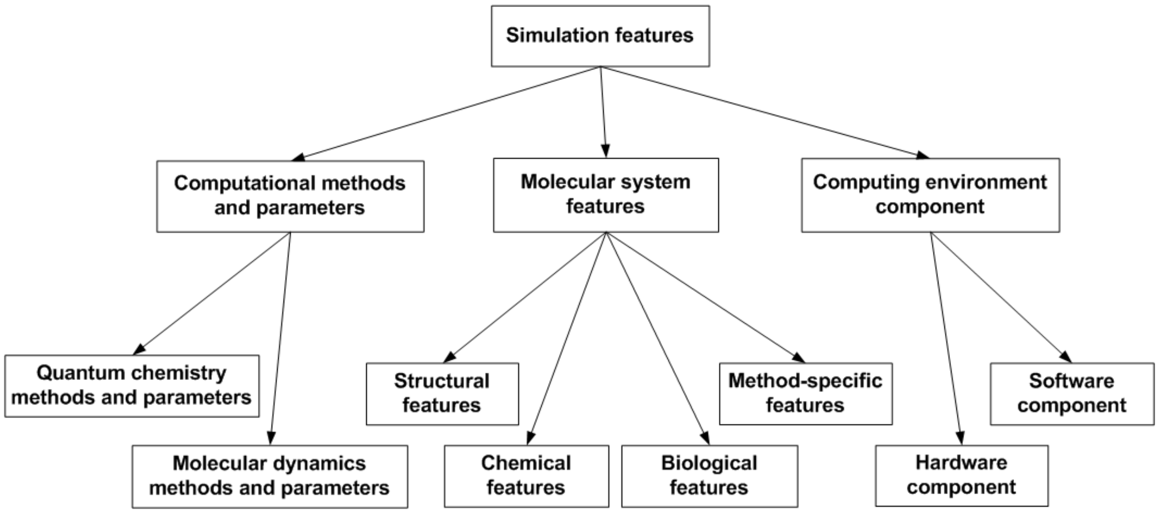
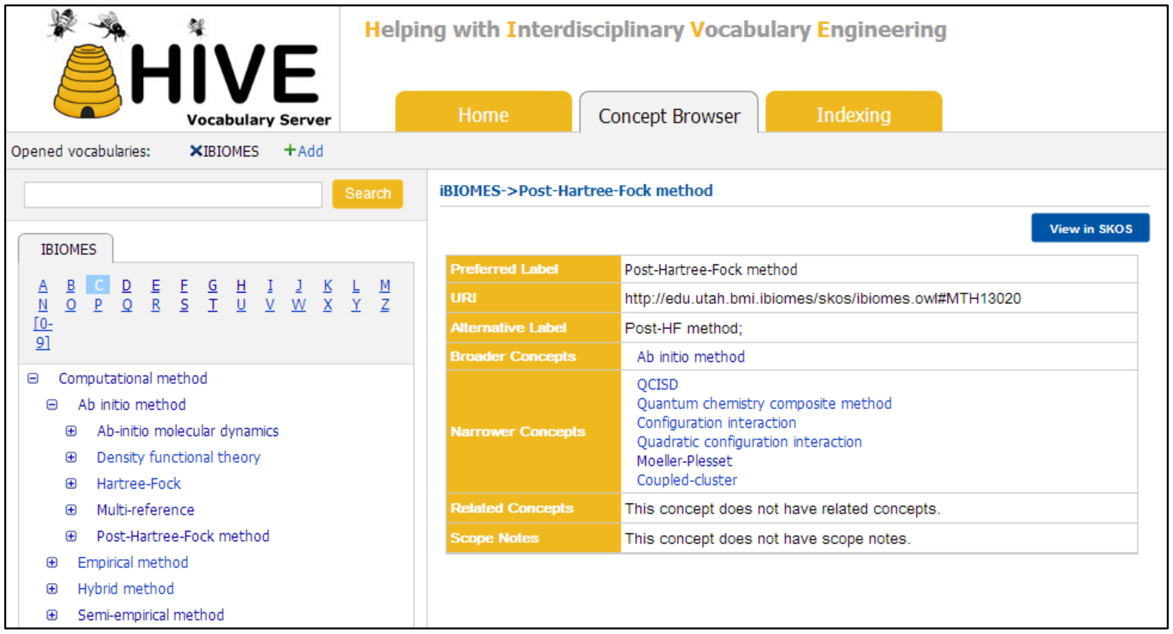

Simulation features
Computational methods and parameters
Molecular system features
Computing environment component
Quantum chemistry methods and parameters
Structural features
Method-specific features
Software component
Molecular dynamics methods and parameters
Chemical features
Biological features
Hardware component

Figure 4.3, Thesaurus semantic network



iBIOMES->Post-Hartree-Fock method

Preferred Label	Post-Hartree-Fock method
URI	http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#MTH13020
Alternative Label	Post-HF method;
Broader Concepts	Ab initio method
Narrower Concepts	QCISD Quantum chemistry composite method Configuration interaction Quadratic configuration interaction Moeller-Plesset Coupled-cluster
Related Concepts	This concept does not have related concepts.
Scope Notes	This concept does not have scope notes.

Figure 4.4, Screenshot of the Hive web interface for SKOS concept browsing

```
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix ib: <http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#> .
@base <http://edu.utah.bmi.ibiomes/skos/ibiomes.owl> .

ib:IBIOMES rdf:type skos:ConceptScheme ;
  rdfs:label "IBIOMES"@en ;
  rdfs:comment "Vocabulary for biomolecular simulations"@en ;
  dc:title "Vocabulary for biomolecular simulations";
  dc:date "2014-03-23";
  dc:creator "Julien Thibault" .

ib:MTH10000 rdf:type skos:Concept ;
  skos:inScheme ib:IBIOMES ;
  skos:prefLabel "Computational method"@en ;
  skos:altLabel "Method"@en ;
  skos:definition "Computational method"@en ;
  skos:narrower ib:MTH11000 ;
  skos:narrower ib:MTH12000 ;
  skos:narrower ib:MTH13000 ;
  skos:narrower ib:MTH14000 .

ib:IBIOMES skos:hasTopConcept ib:MTH10000 .

ib:MTH11000 rdf:type skos:Concept ;
  skos:inScheme ib:IBIOMES ;
  skos:prefLabel "Empirical method"@en ;
  skos:definition "Computational method that uses empirical parameters"@en ;
  skos:broader ib:MTH10000 ;
  skos:narrower ib:MTH11100 ;
  skos:narrower ib:MTH11200 .

ib:MTH11100 rdf:type skos:Concept ;
  skos:inScheme ib:IBIOMES ;
  skos:prefLabel "Classical molecular dynamics"@en ;
  skos:altLabel "Classical MD"@en ;
  skos:definition "Molecular mechanics-based molecular dynamics"@en ;
  skos:broader ib:MTH11000 ;
  skos:narrower ib:MTH11110 ;
  skos:narrower ib:MTH11120 ;
  skos:narrower ib:MTH11300 .
```

Figure 4.5, SKOS document extract in RDF/Turtle format

SPARQL query

```
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX ib: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#>
SELECT ?uri ?label
WHERE {
      ib:MD00900 skos:narrower ?uri .
      ?uri skos:prefLabel ?label
}
```

Output

```
[1] label: "Classical force field"@en
    uri: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#C17501

[2] label: "Polarizable force field"@en
    uri: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#C17502

[3] label: "Reactive force field"@en
    uri: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#C17503

[4] label: "Coarse-grain force field"@en
    uri: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#C17504
```

Figure 4.6, SPARQL query example: retrieving the concepts that are narrower than the ‘Force field’ concept (MD00900).

SPARQL query

```
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX ib: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#>
SELECT ?uri ?label
WHERE {
      ib:MTH11100 skos:broader ?uri .
      ?uri skos:prefLabel ?label
}
```

Output

```
[1] label: "Empirical method"@en
    uri: http://edu.utah.bmi.ibiomes/skos/ibiomes.owl#MTH11000
```

Figure 4.7, SPARQL query example: retrieving the concepts that are broader than the ‘Classical molecular dynamics’ concept (MTH11100).

Table 4.1, Relationship mappings between thesaurus, SKOS, and ontology

Relationship	SKOS equivalents	OWL equivalents
[A] is a [B]	[A] skos:narrower [B] [B] skos:broader [A]	[A] rdfs:subClassOf [B]
[A] has part [B]	[A] skos:relatedHasPart [B]	[A] <http://purl.obolibrary.org/obo/BFO_0000051> [B]
[B] part of [A]	[B] skos:relatedPartOf [A]	[B] <http://purl.obolibrary.org/obo/BFO_0000050> [A]

Table 4.2, Mappings between BIOSIO concepts and external OBO ontologies

OBO parent			BIOSIO children	
Ontology	**Concept**	**Label**	**Concept**	**Label**
BFO	BFO_0000019	Quality	MTH00100	Descriptor
BFO	BFO_0000028	Three-dimensional spatial region	SYS11000	Box
BFO	BFO_0000030	Object	CPE00002 HW01000	Computing platform Hardware component
BFO	BFO_0000031	Generically dependent continuant	PRM00001 PRM00101	Parameter Parameter set
IAO	IAO_0000010	Software	SW01100 SW01200	Operating system Software package
IAO	IAO_0000030	Information content entity	FS01000	File system
IAO	IAO_0000098	Data format specification	FS01110	File format
IAO	IAO_0000104	Plan specification	MTH10000	Computational method
IAO	IAO_0000115	Definition	#citation	Citation
IAO	IAO_0000310	Document	FS01100	File
OBI	OBI_0200000	Data transformation	TSK10000 TSK00001	Computational process Computational task
CHEBI	CHEBI_24431	Chemical entity	SYS10000 SYS01000	Molecular system Molecular system component

Table 4.3, Sample matches between thesaurus concept terms and UMLS concept names.

Concept	Matching term	UMLS CUI	UMLS concept name	UMLS sources	Match
Hartree-Fock	HF	C3273279	CFH wt Allele	NCI	No
		C1538440	CFH gene	OMIM	No
Solvent-accessible surface	SAS	C1426104	NANS gene	OMIM, HGNC	No
		C1823519	TSPAN31 gene	OMIM	No
Protein Data Bank	Protein Data Bank	C1705318	Protein Data Bank	MTH, NCI	Yes
Graphics Processing Unit	GPU	C1881002	Graphics Processing Unit	NCI	Yes
Central Processing Unit	CPU	C1707144	Central Processing Unit	NCI	Yes
		C1413666	CPB2 gene	OMIM, HGNC	No
Volume	Volume	C0449468	Volume	LNC, FMA, NCI, MTH, SNOMEDCT…	Yes
		C1705102	Volume (publication)	NCI	No
Analysis task	Analysis	C0002778	Analysis of substances	SNOMEDCT	No
		C0936012	Analysis	MTH, PSY	Parent

References

1. Dror, R. O.; Dirks, R. M.; Grossman, J. P.; Xu, H.; Shaw, D. E., Biomolecular Simulation: a Computational Microscope for Molecular Biology. *Annu. Rev. Biophys.* **2012**, 41, 429-452.

2. Karplus, M.; McCammon, J. A., Molecular dynamics simulations of biomolecules. *Nature Structural & Molecular Biology* **2002**, 9, 646-652.

3. Alonso, H.; Bliznyuk, A. A.; Gready, J. E., Combining Docking and Molecular Dynamic Simulations in Drug Design. *Med. Res. Rev.* **2006**, 26, 531-568.

4. Durrant, J. D.; McCammon, J. A., Molecular Dynamics Simulations and Drug Discovery. *BMC biology* **2011**, 9, 71.

5. Lane, T. J.; Shukla, D.; Beauchamp, K. A.; Pande, V. S., To milliseconds and beyond: challenges in the simulation of protein folding. *Curr. Opin. Struct. Biol.* **2013**, 23, 58-65.

6. Thibault, J. C.; Facelli, J. C.; Cheatham, T. E., 3rd, iBIOMES: Managing and Sharing Biomolecular Simulation Data in a Distributed Environment. *J. Chem. Inf. Model.* **2013**, 53, 726-736.

7. Ng, M. H.; Johnston, S.; Wu, B.; Murdock, S. E.; Tai, K.; Fangohr, H.; Cox, S. J.; Essex, J. W.; Sansom, M. S. P.; Jeffreys, P., BioSimGrid: Grid-Enabled Biomolecular Simulation Data Storage and Analysis. *Future Gener. Comp. Sy.* **2006**, 22, 657-664.

8. Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Lluis Gelpi, J., MDWeb and MDMoby: An Integrated Web-Based Platform for Molecular Dynamics Simulations. *Bioinformatics* **2012**, 28, 1278-1279.

9. Simms, A. M.; Toofanny, R. D.; Kehl, C.; Benson, N. C.; Daggett, V., Dynameomics: Design of a Computational Lab Workflow and Scientific Data Repository for Protein Simulations. *Protein Eng. Des. Sel.* **2008**, 21, 369-377.

10. Meyer, T.; D'Abramo, M.; Hospital, A.; Rueda, M.; Ferrer-Costa, C.; Perez, A.; Carrillo, O.; Camps, J.; Fenollosa, C.; Repchevsky, D.; Lluis Gelpi, J.; Orozco, M., MoDEL (Molecular Dynamics Extended Library): A Database of Atomistic Molecular Dynamics Trajectories. *Structure* **2010**, 18, 1399-1409.

11. Thibault, J. C.; Roe, D. R.; Facelli, J. C.; Cheatham III, T. E., Data model, dictionaries, and desiderata for biomolecular simulation data indexing and sharing. *J. Cheminform.* **2014**, 6, 4.

12. Goni, R.; Apostolov, R.; Lundborg, M.; Bernau, C.; Jamitzky, F.; Laure, E.; Lindhal, E.; Andrio, P.; Becerra, Y.; Orozco, M.; Lluis Gelpi, J., Standards for Data Handling. *ScalaLife White Paper* **2013**.

13. Hinsen, K., MOSAIC: A Data Model and File Formats for Molecular Simulations. *J. Chem. Inf. Model.* **2013**.

14. Bodenreider, O., The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic Acids Res.* **2004**, 32, D267.

15. Simple Knowledge Organization System (SKOS) reference. http://www.w3.org/TR/skos-reference/

16. OWL 2 Web Ontology Language Document Overview (Second Edition). http://www.w3.org/TR/owl2-overview/

17. McCray, A. T.; Burgun, A.; Bodenreider, O., Aggregating UMLS semantic types for reducing conceptual complexity. *Stud. Health Technol. Inform.* **2001**, 216-220.

18. RDF 1.1 Turtle (Terse RDF Triple Language). http://www.w3.org/TR/turtle/

19. OWL API. http://owlapi.sourceforge.net/

20. SKOS API. http://skosapi.sourceforge.net/

21. Grenon, P.; Smith, B.; Goldberg, L., Biodynamic ontology: applying BFO in the biomedical domain. *Stud. Health Technol. Inform.* **2004**, 102, 20-38.

22. Brinkman, R. R.; Courtot, M.; Derom, D.; Fostel, J. M.; He, Y.; Lord, P.; Malone, J.; Parkinson, H.; Peters, B.; Rocca-Serra, P.; Ruttenberg, A.; Sansone, S. A.; Soldatova, L. N.; Stoeckert, C. J., Jr.; Turner, J. A.; Zheng, J.; consortium, O. B. I., Modeling biomedical experimental processes with OBI. *J. Biomed. Semantics.* **2010**, 1 Suppl 1, S7.

23. Hastings, J.; de Matos, P.; Dekker, A.; Ennis, M.; Harsha, B.; Kale, N.; Muthukrishnan, V.; Owen, G.; Turner, S.; Williams, M.; Steinbeck, C., The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013. *Nucleic Acids Res.* **2013**, 41, D456-63.

24. Ashburner, M.; Ball, C. A.; Blake, J. A.; Botstein, D.; Butler, H.; Cherry, J. M.; Davis, A. P.; Dolinski, K.; Dwight, S. S.; Eppig, J. T.; Harris, M. A.; Hill, D. P.; Issel-Tarver, L.; Kasarskis, A.; Lewis, S.; Matese, J. C.; Richardson, J. E.; Ringwald, M.; Rubin, G. M.; Sherlock, G., Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature genetics* **2000**, 25, 25-9.

25. Natale, D. A.; Arighi, C. N.; Barker, W. C.; Blake, J. A.; Bult, C. J.; Caudy, M.; Drabkin, H. J.; D'Eustachio, P.; Evsikov, A. V.; Huang, H., The Protein Ontology: a structured representation of protein forms and complexes. *Nucleic Acids Res.* **2011**, 39, D539.

26. Gennari, J. H.; Musen, M. A.; Fergerson, R. W.; Grosso, W. E.; Crubézy, M.; Eriksson, H.; Noy, N. F.; Tu, S. W., The evolution of Protégé: an environment for knowledge-based systems development. *Int. J. Hum-Comput. St.* **2003**, 58, 89-123.

27. Apache Lucene. https://lucene.apache.org/

28. *Lexical Variant Generator (LVG) Java API*, 2014 release; 2014.

29. Rajasekar, A.; Moore, R.; Hou, C.; Lee, C. A.; Marciano, R.; de Torcy, A.; Wan, M.; Schroeder, W.; Chen, S. Y.; Gilbert, L., iRODS Primer: Integrated Rule-Oriented Data System. *Synthesis Lectures on Information Concepts, Retrieval, and Services* **2010**, 2, 1-143.

30. Helping Interdisciplinary Vocabulary Engineering (HIVE). https://code.google.com/p/hive-mrc/

31. W3C SPARQL 1.1 Overview. http://www.w3.org/TR/sparql11-overview/

32. Medelyan, O.; Witten, I. H. Thesaurus based automatic keyphrase indexing. In Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries, 2006; ACM: 2006; pp 296-297.

33. Hastings, J.; Chepelev, L.; Willighagen, E.; Adams, N.; Steinbeck, C.; Dumontier, M., The chemical information ontology: provenance and disambiguation for chemical data on the biological semantic web. *PloS one* **2011**, 6, e25513.

34. Cimino, J. J., Desiderata for controlled medical vocabularies in the twenty-first century. *Methods Inf. Med.* **1998**, 37, 394-403.

35. Bright, T. J.; Yoko Furuya, E.; Kuperman, G. J.; Cimino, J. J.; Bakken, S., Development and Evaluation of an Ontology for Guiding Appropriate Antibiotic Prescribing. *J. Biomed. Inform.* **2012**, 45, 120-8.

36. Ananiadou, S.; Mima, H., Automatic recognition of multi-word terms: the C-value/NC-value method. *Int. J. Digit. Libr.* **2000**, 3, 115.

CHAPTER 5

IBIOMES: MANAGING AND SHARING BIOMOLECULAR SIMULATION DATA IN A DISTRIBUTED ENVIRONMENT[1]

Abstract

Biomolecular simulations, which were once batch queue or compute limited, have now become data analysis and management limited. In this paper we introduce a new management system for large biomolecular simulation and computational chemistry datasets. The system can be easily deployed on distributed servers to create a minigrid at the researcher's site. The system not only offers a simple data deposition mechanism but also a way to register data into the system without moving the data from their original location. Any registered dataset can be searched and downloaded using a set of defined metadata for Molecular Dynamics and Quantum Mechanics, and visualized through a dynamic web interface.

[1] Reproduced in part with permission from Thibault, J.C., Facelli, J.C., and Cheatham III, T.E. (2013). iBIOMES: Managing and Sharing Biomolecular Simulation Data in a Distributed Environment. Journal of Chemical Information and Modeling, 53(3), 726-736. Copyright 2014 American Chemical Society.

Introduction

Biomolecular simulations aim to study the structure, dynamics, interactions, and energetics of complex biomolecular systems. Understanding biological phenomena with these methods may facilitate the design of better drugs, therapies, catalysts and nanotechnology.[1-3] With the recent advances in hardware, it is now not only possible to use more complex and accurate models, but also to reach time scales that are biologically significant. When simulating biomolecular dynamics on the microsecond time scale for example, one can easily generate molecular dynamics trajectories of the time series of atomic positions that represent terabytes (TB) of data on disk. More recently, special-purpose hardware such as the Anton machine has allowed researchers to reach millisecond time scales,[4] increasing the size of the resulting data even further. While the computing power has dramatically increased in the last decade, our ability to manage, store, analyze, and move large datasets is still limited. Central repositories for the community or even at the lab level are desirable to facilitate data management, analysis, and sharing. This will require both new methods to catalog existing datasets by keeping them in place and improved mechanisms for facilitating and cataloguing data storage and movement.

Biomolecular simulations and computational chemistry are dominated by two classes of methods: Molecular dynamics (MD) and quantum mechanics (QM). Many variations (based on parameter choice or approximations) of the methods exist, along with hybrid approaches that combine different methods. A wide variety of MD and QM codes are available to the scientific community. AMBER,[5] NAMD,[6] CHARMM,[7] GROMACS,[8] and LAMMPS,[9] are some of the most popular MD simulation codes in use today to simulate proteins, nucleic acids, or even larger molecules. Gaussian,[10] NWChem,[11] GAMESS,[12] Q-Chem,[13] Jaguar,[14] and VASP[15] on the other hand, are popular QM packages, typically used to study small molecules such as drug compounds. The heterogeneity of the data resulting from the simulations (e.g., QM calculation vs. MD atom trajectories), and the format of input and output files makes data management non-trivial. Moreover, each simulation software package has its own way to represent simulation parameters (e.g., simulated time, method), molecule topologies, and resulting data (e.g., trajectories of the times series of atomic positions). Additionally, each lab has multiple researchers (including students, post-docs, staff) using local and national resources, different software packages and methods, different file naming conventions, and different analysis workflows. As a result it can become quite complicated for investigators to manage this distributed multiuser environment and retrieve summaries of simulations that were run in the past.

The heterogeneity of biomolecular simulation data and the distributed nature of the resources used by researchers become even more obvious as we move towards collaboration between labs, and across institutions. Nevertheless, sharing data outside the owner’s institution has a scientific purpose. As theoretical models (e.g., basis sets, force-fields) and implementations evolve developers need to validate their code by comparing results to existing implementations. Creating collaborative networks for developers of a particular software package would increase the number of testing and validation datasets available to them. For biomedical researchers, the more datasets become available to the community, the easier it is to expose correlations between experiments and provide insight into biological structure and function. A successful example is the ABC (Ascona

B-DNA Consortium) initiative, led by multiple laboratories distributed all over the world. A large series of MD simulations of B-DNA were run by the many groups in a divide-and-conquer manner to expose sequence-specific nucleic acid structure and dynamics.[16-19] A significant challenge has been to aggregate the data. Such initiatives could be facilitated if labs had tools to manage and share their data within a collaborative network or with the community at large.

Sharing raw simulation data with the community would also facilitate replication of results and increase the trustworthiness of related publications. For a single software package, there might be hundreds of different parameters a user can set, and related publications typically will not include all of them. Replication of a simulation run will then require guesses if the original input files are not made publicly available. Finally, there may be unanticipated uses of MD data that will prove community-level databases to be desirable (e.g., the development of coarse-grained force fields parameterization or novel analyses of the existing data).

Because of the amount of data researchers have to deal with, it is not always practical to centralize the data for collaboration. Distributed systems offer a good solution for scientific research in general. Distributed data sources can be aggregated as a single resource despite being physically distant, and local control over the data at each node can be conserved. This is very important as researchers tend to be reluctant to expose all their data or give up ownership. Distributed systems, such as the Grid,[20] allow researchers to keep control over their own data (storage, backup, security) while offering the tools to expose them to the community with authentication and authorization mechanisms.

Although data management systems at the community level are important, new mechanisms are needed to facilitate or even automate the integration of local data owned by individual researchers into collaborative or public repositories. While local data are usually unorganized (file system versus database) and dynamic by nature, public repositories tend to be more static and more structured to enable domain-specific queries by researchers. Mapping these two approaches seamlessly is not a trivial task. Three levels of granularity for data management should be considered. First, at the lowest level, tools should provide a means for individual researchers to effectively catalogue, browse, and search their data, and expose features across datasets. In the case of MD simulation data, such features might include, beyond the raw simulation data and input files, summaries of the analysis such as root-mean-squared deviation (RMSD) plots versus time, molecular graphics of average structures, and/or sequence/topology information. The tools used to catalogue and collect these data should not be onerous or complicated. They also need to run in closed environments where the data owner might not have root privileges (e.g., national computer resources). Finally, data presentation should be customizable so that the user can specify which analysis results should be considered for display to summarize a particular experiment. At the next level, data management tools should allow users to share information (and customizations) within their group or lab. Ultimately, these tools should allow users to share their data with the community either by granting access to their existing data in a secured fashion or by copying the data and its description (i.e., the metadata) to a public repository.

An important aspect of biomolecular simulation data management is the ability to catalogue the data not only at the level of an individual simulation – typically physically represented by a single set of files or a single directory of data on a file system – but also across larger experiments or projects distributed among multiple file systems and directories of data. In the context of this work we consider an experiment or project as a set of dependent QM or MD runs. For example MD experiments usually require a minimization and an equilibration preprocessing phase. Here the minimization-equilibration-production runs would be considered as a single experiment. Experiments can be grouped together to form experiment sets, for example, independent runs of a similar system with different force fields or simulation protocols (i.e., related but independent simulations, results and files). By providing organization not only at the level of individual simulations but across related experiment sets, the user is provided with a greater ability to manage and search physical data (files and directories) and logical sets.

In this paper, we introduce iBIOMES (integrated BIOMolEcular Simulations), a distributed system for biomolecular simulation data management. Input and output files can be easily registered into the system and indexed using a set of metadata, automatically generated by format-specific parsers. Servers containing existing datasets can be easily integrated into the system to avoid large data movements and still benefit from the indexing capabilities of iBIOMES. A prototype is deployed at the University of Utah and is being developed to expose a subset of the MD and QM datasets generated by our lab over the years. Data are managed via a Java API and exposed via a web portal (http://ibiomes.chpc.utah.edu).

Several projects have tried to tackle the problem of molecular simulation data storing and/or sharing. We can distinguish two types of infrastructure: one that is purely based on relational databases, and one that keeps references to the raw input and output files and only stores simulation metadata in a relational database. The BioSimGrid project[21] and the Dynameomics project[22] belong to this first category, where trajectory information is stored directly into database tables, using one entry for each atom and for each time frame. Scalability of pure relational databases using this approach becomes problematic as we reach larger molecular systems and biologically-relevant time scales. For example, in our lab we have over 200 TB of raw MD simulation data including multiple microsecond scale simulations containing millions of frames of trajectory data; replicating the raw data into a database is impractical, wasteful of disk resources, and would be extremely slow to process. Another issue for these databases is the lack of analysis tools as most current analysis tools perform their calculations on the raw files, and not on database tables. The eMinerals project[23,24] and the MoDEL (Molecular Dynamics Extended Library)[25,26] databases adopted a different approach where the raw output files (or a compressed version) are made available and searchable through a database that stores information about the runs (e.g., PDB ID, molecule name). The advantage of keeping the raw files is that it becomes easier to replicate the results if necessary and existing tools can be used to perform the analysis of trajectory files.

For the iBIOMES project, we designed and implemented a distributed solution to data storage and sharing across research labs using this second approach. Simplicity was one of the key concerns for the development of this system. Users should be able to deposit, search, and retrieve data into and from the system easily through simple commands, similar to those offered by the Bookshelf system.[27] The iBIOMES system provides such a command-line interface along with a web interface which offers extra visualization components. Another key concern was the ability to deploy the system locally without interfering with the lab workflow. Data can be “deposited” into the system – i.e., copied from a remote resource to a resource that is part of the system – or simply “registered” in place if the host server is integrated into the system. This becomes a crucial necessity as labs tend to have multiple servers storing terabytes of data and moving these data to be tracked by the system is not practical. The underlying data handling system, based on the iRODS (Integrated Rule Oriented Data System) framework,[28] creates a virtual data warehouse at the researcher’s site, where data can be distributed among multiple servers and searched through metadata query. Metadata include system information (e.g., file location, file name, permissions, registration date) and iBIOMES-defined metadata (e.g., simulation description, title, force field used) that are used to index MD simulations or QM calculations. iRODS provides a command-line interface to manage all the servers and the files that are registered into the system. iBIOMES offers several other commands that are used to publish simulation files into the system and automatically generate metadata. A web portal and a REST (REpresentational State Transfer[29]) interface are also available to facilitate queries of MD and QM data for the end-user and external systems. In the next sections, we will give more details about the iRODS data-handling system, the metadata being used, and the different user interfaces that were specifically developed for iBIOMES.

The iRODS data-handling system

The Integrated Rule Oriented Data System (iRODS)[28] is a file management system that provides the tools to register, move, and lookup files that are distributed over the network and stored in different types of disk (e.g., HPC servers, files servers, archive tapes). iBIOMES uses iRODS as its underlying data handling system to manage distributed resources. Files that are registered into an iRODS zone are accessed using a virtual path that hides the physical location of the files (and servers), which makes it simple for users to logically organize their own data in a distributed environment. Information about the resources and the files registered into an iRODS zone are stored into the iCAT (iRODS CATalog) database. This database keeps track of the system information (e.g., file location, file name, owner) and user-defined metadata that allow any triplet “attribute, value, unit” (AVU). A simplified example of a user metadata table is given in Table 5.1. User-defined metadata can be used to search and retrieve distributed data that are registered in iRODS.

A command line interface is available to manage this virtual warehouse. The “i-commands” provide the necessary functionalities one would need in a Unix-like environment to move data between servers, manage file permissions, users and groups, etc. Commands are also available to check data integrity, i.e., whether a registered file physically exists and if its content has not been altered outside iRODS. The *ifsck* command can be used to compare the size or checksum of the physical file with its corresponding entry in the system, while the *iscan* command can parse the file system to check if a physical file or directory is already registered into iRODS. iRODS also provides a powerful rule engine to manage policies and respond to specified conditions (e.g., registration of a new file) by applying a defined rule (e.g., synchronize the file with another server). Command-line and web interfaces are provided to lookup files based on user-defined metadata or system metadata. iRODS is supported by the Data Intensive Cyber Environment (DICE), which is also responsible in part for the development of the

Storage Resource Broker (SRB).[30] Although SRB is still supported, iRODS became the DICE-recommended framework to manage distributed data. Several national and international scientific projects have already successfully adopted iRODS for their cyberinfrastructure needs. The Wellcome Trust Sanger Institute and the Broad Institute currently use iRODS to manage sequencing data.[31] The iPlant Collaborative project[32] uses iRODS to manage data gathered from all plant sciences, including genotypic and phenotypic data. iRODS has also been used to manage astronomy data, typically images in the gigabyte range (National Optical Astronomy Observatory (NOAO), International Virtual Observatory Alliance (IVOA)). National computational Grids have also started to use iRODS for data management in their widely distributed environments. XSEDE (Extreme Science and Engineering Discovery Environment, https://www.xsede.org), a large cyberinfrastructure project in the US, now offers data replication services based on iRODS at a number of its sites (e.g., National Center for Supercomputing Applications, Pittsburgh Supercomputing Center, Texas Advanced Computing Center). The Open Science Grid (OSG) is following the trend and is currently integrating iRODS into their cyberinfrastructure (www.opensciencegrid.org). This adoption by major computational centers is very important. First it creates a strong community of users and developers. Then it facilitates the federation of remote sites together, and therefore the deployment of systems such as iBIOMES to fulfill the needs of scientists in a particular area. While iRODS provides generic data and metadata storage and query capabilities, iBIOMES offers a domain-specific metadata catalog and customized user interfaces for biomolecular simulation data.

iBIOMES architecture

The general architecture of iBIOMES is presented in Figure 5.1. At the lowest level, iRODS stores the file/collection metadata in a PostgreSQL database (http://www.postgresql.org), and provides interfaces to manage the distributed resources integrated into the system. A MySQL database (http://www.mysql.com) was added to store MD and QM related metadata definitions and dictionaries such as lists of force-fields, basis sets, software, and definitions of experiment sets. Each experiment set can be assigned a name, description, and a set of metadata. While each experiment is assumed to be a physical directory somewhere in the system, sets are logical groups of experiments where each experiment can be part of multiple sets. A Java API (iBIOMES-core) was created to programmatically access iRODS resources and to manage metadata that are specific to biomolecular simulations. The API also helps to generate metadata by parsing the files that are being registered into the system in order to avoid manual annotation by the data owner. Access to iRODS functionalities is facilitated through the Jargon Java API provided by iRODS. Finally, a RESTful interface and a web portal provide access to the registered data in a more user-friendly fashion.

Metadata

When working with biomolecular simulation data, several pieces of information are needed to summarize and index the experiments. Our current list of metadata covers the following categories: authorship (e.g., owner, related publications), methods (e.g., MD or QM, basis set, force field, parameters), molecular system (e.g., topology, type of molecule), platform (hardware and software information), and files (e.g., format). Our goal is to develop a list of core metadata that would be software-independent, and sufficient to retrieve raw data files that contain the necessary details to replicate an experiment. The metadata schema database contains the current list of metadata attributes and their definitions. A subset of the metadata attributes defined in iBIOMES is given in Table 5.2. This database also contains several dictionaries such as lists of force fields, basis sets, or software packages that users can use to facilitate queries or annotations of experiments. This list is extensible and allows custom user-defined metadata.

The distinction between experiment and experiment set is important when registering data into iBIOMES. Metadata are automatically generated for the files through the API's parsers then pushed up to the experiment level. For example, in a directory containing AMBER simulation data, the topology-related metadata are parsed from AMBER topology files, or PDB files if not available. The new topology metadata set is then added to the root directory, which is considered to be the representation of the experiment. Currently, no metadata are generated for experiment sets, but the owner can easily pick one of the experiments or a file to push metadata to the experiment set level. For example if the topology information is the same for all experiments within the set, this information can be easily pulled and applied to the set level via the web interface.

Currently, automatic metadata generation is supported for PDB files, MOL/SDF files, Mol2 files, AMBER topology, input, and output files, GROMACS Include Topology (.itp), System Topology (.top), and parameter input (.mdp) files, Protein Structure Files (.psf), NWChem, Gaussian, and GAMESS input files. Each parser implementation is based on the conceptual model summarized in Figure 5.2.

File parser classes inherit from AbstractTopologyFile, AbstractParameterFile, or AbstractParameterAndTopologyFile, whether the target file format defines topology information, calculation parameter, or both. For example the Gaussian input file parser inherits from AbstractParameterAndTopology since it needs to parse the QM calculation parameters (e.g., basis set, level of theory) and the compound topology, while the PDB parser only looks at topology information and inherits from AbstractTopologyFile.

In order to implement a new parser one needs to create a new Java class that inherits from one of the abstract classes and write a parsing function that will build the Method and/or MolecularSystem (i.e., a set of molecules) objects. Mapping between this data model and the iBIOMES metadata is done through the getMetadata() method available for each of the classes inheriting from Method and Molecule. This method is automatically called when registering the files into iBIOMES.

While in most cases rules for parsing files can be applied solely based on the file name extension (e.g., .pdb), there are cases where the format of a file cannot be determined based on its extension. To overcome this issue and enable automatic metadata assignment and extensibility, a set of rules can be defined in an XML descriptor file. Rules can define metadata for files or directories with names matching a specified pattern. Examples of such rules are given in Figure 5.3. In this example the first rule defines possible file extensions for AMBER topology files (.prmtop, .topo, .top, or .parm). The second rule targets files that are the result of an MD trajectory clustering algorithm. The clustering tool generates averaged structures in PDB format but omits the .pdb file extension. By applying this rule these files are recognized as PDB files when registered into the system and viewable as 3D structures. The last rule targets a CSV (comma-separated value) file that represents a time series, generated by an analysis script. As the same script and name conventions are used in our lab, this rule helps define the labels (e.g., Time, Density), titles (e.g., Evolution of density over time), and units (e.g., ps, g/cm3) for the data contained in the file. Once registered, this file can be automatically displayed through the web interface as a 2D plot with the correct legends and axis titles.

This rule set can be customized to fit the needs of a particular lab or user. Experience showed that file name convention for a particular software package run (e.g., AMBER) and the following analysis vary only slightly for the same user. Therefore the XML file will be reusable. Once a simulation and its associated files are registered into iBIOMES, the owner or the authorized users can still edit the metadata through the web interface (or any iRODS interface).

Interfaces

Web interfaces

A REST interface was developed to offer web services for access to the metadata catalog and dictionaries. The metadata catalog is open access as it only contains general definitions of biomolecular simulation related metadata. The related services are mainly used to auto-complete user entries in the web interface (e.g., software name, force field). The current web portal builds upon this REST interface and allows authenticated and authorized users to manage and search data registered in iBIOMES (Figure 5.4 and 5.5). Users can create queries based on the standard metadata catalog to retrieve simulations of interest. The queries can either target files, experiments (collections of files), or experiment sets. A simple web interface is available to query data files and experiments based on common attributes such as methods, molecule type (e.g., DNA, RNA, protein) or residue chain (nucleotide or amino acid sequence). Residue chains are normalized and used as file or experiment metadata, along with the software-specific residue chains. The normalized residue chains are sequences of 1-letter nucleotide or amino acid codes. For example one could search for a particular protein / RNA system using the following AVUs:

```
RESIDUE_CHAIN_NORM = "%GGCUCGUGUAGCUCAUUAGCUCCGAGCC%"
RESIDUE_CHAIN_NORM = "%SGPRPRGTRGKGRRIRR%"
```

Or using AMBER-specific residue chains:

```
RESIDUE_CHAIN = "%RG5 RG RC RU RC RG RU RG RU RA RG RC RU RC RA RU RU RA RG RC RU RC RC RG RA RG RC RC3%"
RESIDUE_CHAIN = "%SER GLY PRO ARG PRO ARG GLY THR ARG GLY LYS GLY ARG ARG ILE ARG ARG%"
```

Although the first approach enable searches through experiments generated by different software packages, the second approach is still useful as certain residue codes are meaningful only in the context of a particular software package or within a community.

Experiments can also be retrieved by simply entering keywords, in which case the metadata attribute is bypassed and the query only uses the value component of the AVU triplets to find matches. Advanced queries can be built as well. The user can pick and choose metadata attributes from the iBIOMES metadata catalog or manually enter user-specific attributes, then assign values to each attribute. Figure 5.6 shows how one could build a query through the web interface using the catalog of standard iBIOMES metadata.

Matching experiments and files can be downloaded and data content can be summarized directly through different applets if the user has the right permissions. For example Jmol[33] is used for 3D rendering of molecules described in PDB, Mol2, MOL/SDF or Gaussian log files (Figure 5.7). Users can pick Jmol-supported files and load them into the applet to compare structures or create multiframe animations. Two-dimensional data such as time series in comma-separated or tab-delimited value format can be dynamically plotted through a service based on the *JFreeChart* (http://www.jfree.org/jfreechart/) library (Figure 5.8a-b). Supported graphs include multiline plots (e.g., comparison of RMSd of multiple runs), scatter plots, and heatmaps (2D-RMSd matrix). A "shopping cart" based on DICE's iDrop applet (https://code.renci.org/gf/project/irodsidrop) also allows users to pick and choose files or collections of files they want to download in a bulk fashion (Figure 5.9).

Experiment sets can be created through the web interface as well. Set owners can define the list of referenced experiments and metadata for a particular set directly from the corresponding experiment set summary page. Experiment sets can be made public or private.

More options are available to experiment data owners or users with write permissions. For example they can manage permissions at the collection or file level and update the associated metadata. iBIOMES-defined metadata can be easily edited using the available dictionaries. User-defined metadata that are not defined in the iBIOMES catalog can be added as well, and used to build queries. While metadata are automatically generated during data publication into the system, the set of metadata might be incomplete or not totally accurate. The web interface allows the user to update topology-specific metadata or method-specific metadata by specifying which files should be used as templates. In the case of the topology for AMBER data, this could be a topology file or a PDB file; for the methods, this could be an MD input or output file. Finally, the main page for a particular experiment can be customized by specifying which 3D structures should be displayed, and which files should be presented to summarize the results. Related publications and published structures (e.g., from the Protein Data Bank,[34] PubChem,[35] or the Cambridge Structural Database[36]) can be added as well for reference.

The web portal was built with Java Server Pages (JSP) and Spring MVC (http://www.springsource.org/). This code, along with the main Java API (iBIOMES-core) was integrated into Maven (http://maven.apache.org/) to manage external dependencies and automate builds.

Data registration

One of the goals of iBIOMES is to make the data publication process as easy as possible. Two scenarios are supported: registration of data into the system without moving the files, and registration after data transfer from a local or remote resource (e.g., desktop, remote computational resource) to an iBIOMES node. Both registration options are available through Unix-like commands that can be run from the machine where the data reside. For in-place registration, the host needs to be integrated to the target iBIOMES zone. Usage of these commands is given in Figure 5.10.

Deployment at the University of Utah

iBIOMES installation requirements

iBIOMES requires a Java Runtime Environment (1.7) to be installed on the host machine. iBIOMES-core is packaged into a single JAR (Java ARchive) file including all the dependencies (e.g., iRODS Java API). As iBIOMES is dependent on iRODS, iRODS should be installed first on the servers that need to be integrated to the system, then the iBIOMES-core library and scripts can be copied on these machines. To host the web application, a web server such as Apache Tomcat (http://tomcat.apache.org) is required to deploy the iBIOMES-web and iBIOMES-ws codes, which are packaged as two WAR (Web application ARchive) files.

iRODS configuration

The current iBIOMES setup for our lab is presented in Figure 5.11. Although all the components of iBIOMES could be installed on a single physical server, we decided to deploy the system in a distributed environment to assess a more likely scenario where data need to be scattered among multiple disks. The primary iRODS server along with the iCAT database were installed on a Linux server (CentOS 5.8). Two file servers (Red Hat Enterprise Linux Server 6.3) were integrated into the same iRODS zone ("ibiomesZone") to provide over 10 TB of disk space overall. Each file server runs an iRODS server instance, and each disk on the servers is exposed as an iRODS resource. Resources can be grouped together to apply data storage policies managed by iRODS. For example one could define a policy to enforce data replication on all resources of the same group, or to order resources in the group to define which resource should be used for storage first. For our case, the 5 resources (5 disks in 2 separate servers) were grouped together and managed through a load balancing policy defined in iRODS. A rule periodically triggers the activation of a resource monitoring system and calculates the load factor on each machine. The iRODS administrator can customize the way the load factor is calculated by assigning a weight to the disk space resource, the CPU load, the memory load, etc. The administration of iRODS servers (start/stop, resource definition, rule control) is made simple through the i-commands and other scripts that can be run only by an iRODS administrator.

iBIOMES deployment

An Apache Tomcat 7 server was installed on the first server to host the web portal and the REST services. The iBIOMES metadata schema database (MySQL) was installed on a second Linux server (CentOS 5.8). This was done through a set of SQL scripts that create the database schema and populate the biomolecular simulation metadata catalog and the dictionaries. The iBIOMES client tools (scripts and JAR file) can be copied to remote resources (e.g., HPC facility) by users to enable data transfer and registration into the system directly from resources outside the defined iRODS zone.

Data summary

Our lab currently owns over 200 TB of both MD simulation and QM calculation datasets. For this prototype we decided to expose a subset of these data that would still be representative of the type of simulation that is done in our lab. Our current projects involve mainly nucleic acid force field developments and P450 QM studies. This is reflected in the datasets currently published in our iBIOMES instance, which for now contain MD simulations of RNA for force-field assessment (AMBER FF 10), and QM calculations that were performed in Gaussian 03 to generate AMBER-compatible heme parameters for various states of the P450 cycle.[37] Because of licensing restrictions, our Gaussian datasets could not be released for public access yet. On the other hand a series of MD simulations of RNA was released, along with a subset of the data derived from the

ABC consortium's study on B-DNA.17 The ABC set currently includes a series of experiments with final stripped trajectories (~20-60 GB each) and basic analysis data (e.g., RMSd, radial plots).

A guest account was created to enable read access for anybody interested in these public datasets. Guests can search experiments, read summaries, and graphically visualize data from this subset. Currently the shopping cart service for bulk downloads is not available for guest logins. Guests can still download files individually. The iBIOMES prototype can be accessed via the guest login option at: http://ibiomes.chpc.utah.edu.

Discussion

In this paper we presented a new distributed system developed to manage large biomolecular simulation datasets. The underlying data handling system based on the iRODS framework creates a virtual data warehouse at the researcher's site, where data can be distributed among multiple servers. Both iRODS and iBIOMES are easy to deploy through a set of scripts. Existing archive servers can be integrated into iBIOMES without a need for a physical reorganization of the files, saving the cost of moving terabytes of data. The current implementation of iBIOMES uses the native iRODS password mechanism to authenticate users. iRODS also supports the Grid Security Infrastructure (GSI) which will facilitate the integration of iBIOMES into scientific Grids. Support for LDAP has been recently added as well. The burden of creating and maintaining iRODS-specific accounts can then be avoided by system administrators, who in turn can deploy iRODS in closed environments with existing security mechanisms and user accounts.

The publication process is facilitated by parsers that automatically generate metadata during file registration, and can be customized for the need of a particular user or lab through XML descriptors. Although our efforts have mainly focused on supporting AMBER and Gaussian datasets, we are currently working on improving our parsers for other popular MD and QM software packages, including GROMACS, CHARMM, Gaussian, GAMESS, and NWChem. Experiments registered into iBIOMES can be easily retrieved through simple keyword searches or queries built upon data elements defined in a metadata catalog for MD simulations and QM calculations. We are currently gathering feedback from the community to define a list of core metadata that would be sufficient to search and retrieve simulation datasets. A data model will be designed to define relationships between the concepts represented by these metadata, and facilitate future semantic integration with external systems, such as scientific grids. In order to enable researchers outside the field of computational chemistry to query data in a meaningful way, it will be necessary to facilitate the annotation of experiments using biological metadata (e.g., molecule name, organism). Currently this type of metadata would have to be entered manually via the web interface after data publication. This process could be facilitated in the future through a web service that would query common databases such as the Protein Data Bank to automatically generate these data elements based on the PDB ID.

Metadata are represented by AVU triplets that can be either tied to the iBIOMES metadata catalog, or customized to represent concepts that are specific to a user or a lab. This provides a very flexible data annotation model compared to a standard relational database schema, where model modifications require an intervention from the database administrators. One limitation of the AVU model is the lack of relations between AVUs. For example, one cannot assign properties to two different molecules (e.g., name, type, residue chain) represented in the same experiment, as attribute names will be the same for both molecules, and cannot be distinguished, as shown in the following example:

```
MOLECULE_TYPE = "RNA"
RESIDUE_CHAIN = "GGCUCGUGUAGCUCA…"
MOLECULE_TYPE = "Protein"
RESIDUE_CHAIN = "SER GLY PRO ARG PRO ARG…"
```

In the current implementation of iBIOMES relations between AVUS cannot be determined. While this is not required for indexing purposes, this becomes necessary to provide a clear conceptual view of the data to the users. To create a more structured metadata schema the iCAT database can be extended with custom tables and enable queries on these tables via the standard iRODS interfaces. Such capability could help us keep track of metadata in a more structured way, especially for multimolecule systems and experiments based on multiple runs using different methods.

The current prototype deployed for our lab demonstrated the ability of iRODS and iBIOMES to manage large biomolecular simulation datasets in a distributed environment. The iBIOMES web portal provides a rich and dynamic user interface to search, download, and visualize data registered into the system. Advanced features are available for data owners to manage permissions, annotate experiments, and customize data display in the web interface. Direct data analysis via iBIOMES is currently not supported. The analysis output has to be explicitly registered into the system and described via metadata to enable visualization through Jmol or the plotting service. This can be achieved automatically by customizing the XML rule set descriptor before data publication or directly via the web interface after data deposit. Thanks to these features users can easily extend the web interface to include new pictures, spreadsheets, or links to any type of data file. The current focus of iBIOMES is not to enable deep analysis of the derived data but instead to provide the means to display, catalogue and share information about biomolecular simulations. As we move forward the system will be enhanced to add simple analysis support (e.g., RMSd calculations, data extraction from time series datasets). Our long-term goal is to provide a complete framework where data can be tracked locally, analyzed via automated processes, and registered seamlessly into a global system such as iBIOMES. For now we hope to learn more from the current iBIOMES system, and define more clearly the needs of the users, such as:

- Which data elements are required or missing for indexing and search purpose?
- How would users interact with iBIOMES to execute complex analysis workflows?
- What can be improved to facilitate education, networking or collaboration between users?

Conclusion

iBIOMES is a new distributed system for biomolecular simulation data management. The data registration process is simple and supported by metadata generators, customizable by the user if needed. Registration does not require physical transfer of the data, which makes it a great solution for researchers who want to expose existing datasets. Finally data summarization and management are facilitated through a rich web interface that offers different visualization components for 3D structures and analysis data (e.g., time series). Guest access to our web portal is currently available at http://ibiomes.chpc.utah.edu.

With the adoption of iRODS across the world, and across scientific domains, we believe that iBIOMES has a strong potential to create collaborative networks within the field of biomolecular simulation, for users, developers, and newcomers to the field.

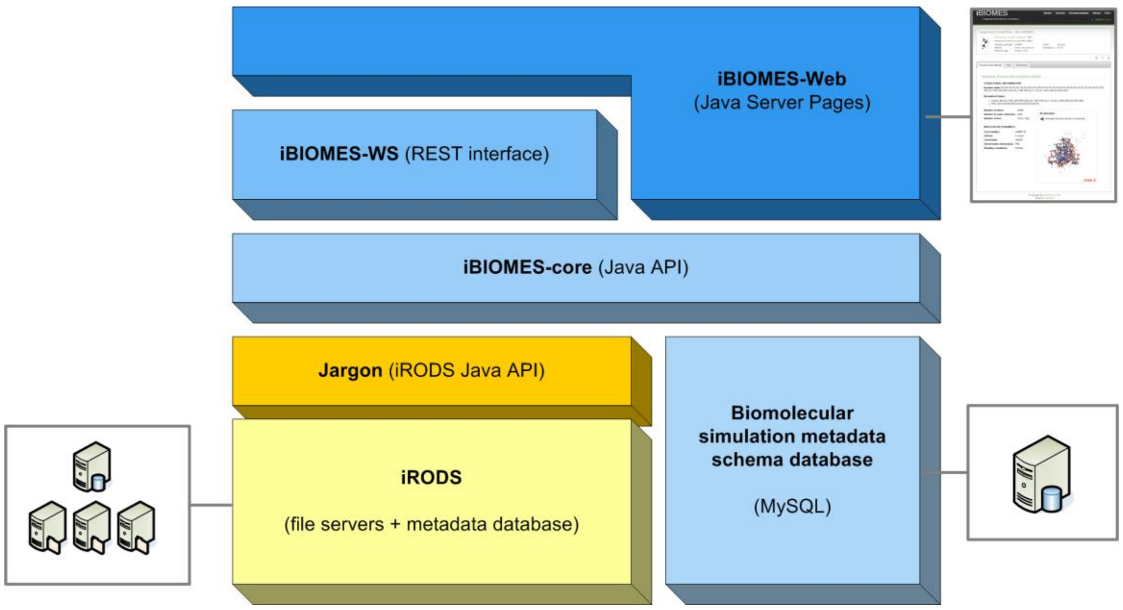

iBIOMES-Web
(Java Server Pages)
iBIOMES-WS (REST interface)
iBIOMES-core (Java API)
Jargon (iRODS Java API)
iRODS
(file servers + metadata database)
Biomolecular simulation metadata schema database
(MySQL)

Figure 5.1, General architecture of iBIOMES. At the lowest level, iRODS stores the file metadata while a separate MySQL database enforces standard metadata use and allows definitions of experiment sets. A REST interface and a web client provide query and update capability to the metadata catalog through the iRODS API (Jargon) and an iBIOMES-specific API (iBIOMES-core).

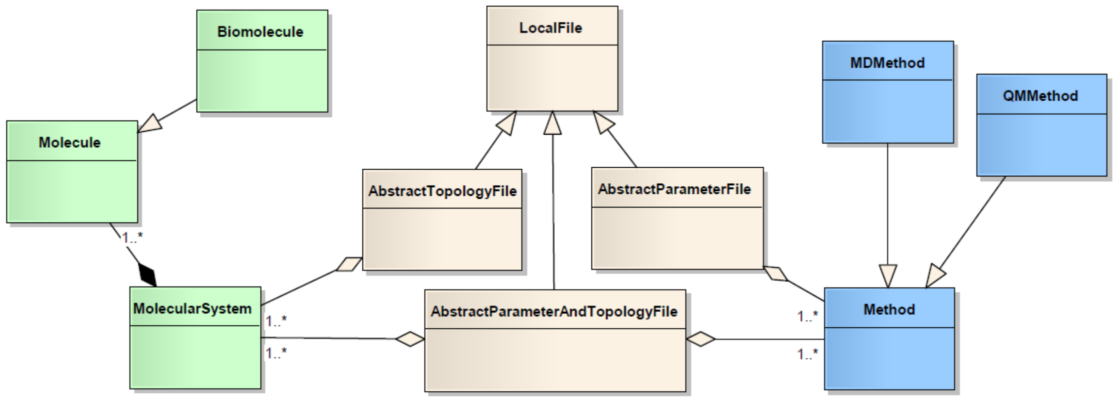
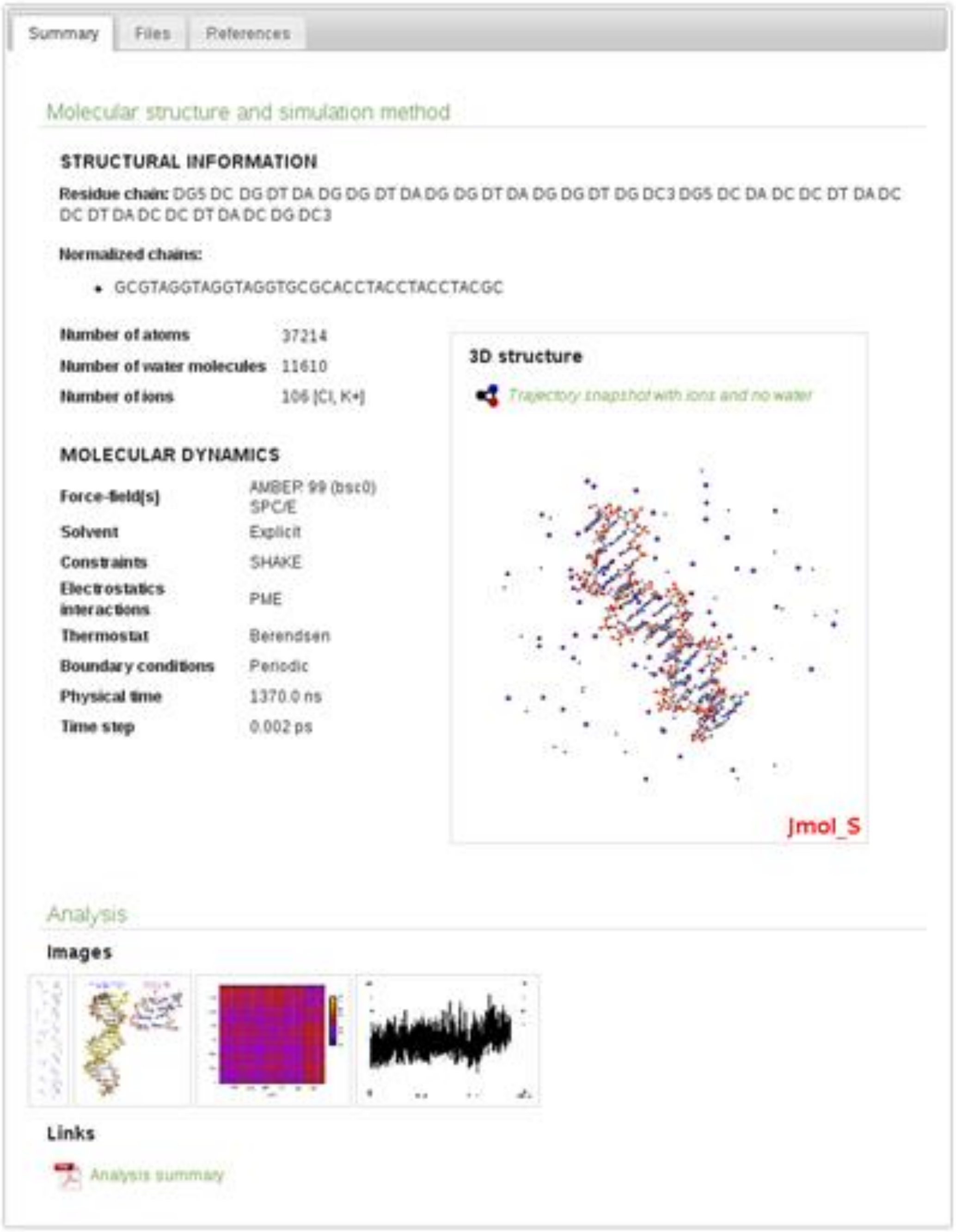


Figure 5.2, Simplified class diagram representing the file parser implementations

```xml
<rules>
  <rule type="file" match="*.(prmtop|topo|top|parm)">
    <metadata>
      <avu attribute="software">AMBER</avu>
      <avu attribute="file_format">AMBER parmtop</avu>
    </metadata>
  </rule>
  <rule type="file" match="*cluster.avg.c*"
        class="analysis_result">
    <metadata>
      <avu attribute="description">
        Averaged structure based on clustering
      </avu>
      <avu attribute="software">ptraj</avu>
      <avu attribute="file_format">PDB</avu>
    </metadata>
  </rule>
  <rule type="file" match="summary.DENSITY(.csv)?"
        class="analysis_result">
      <metadata>
        <avu attribute="description">
          Evolution of density over time
        </avu>
        <avu attribute="data_labels">Time,Density</avu>
        <avu attribute="data_units">ps,g/cm^3</avu>
      </metadata>
    </rule>
</rules>
```

Figure 5.3, Example of XML rule set used to customize the publication process. The first rule associates file extensions to a particular file format (AMBER topology). The second and third rules associate a particular set of metadata to analysis output files that follow a standard nomenclature in our lab.



Summary | Files | References

Molecular structure and simulation method

STRUCTURAL INFORMATION

Residue chain: DG5 DC DG DT DA DG DG DT DA DG DG DT DA DG DG DT DG DC3 DG5 DC DA DC DC DT DA DC DC DT DA DC DC DT DA DC DG DC3

Normalized chains:

- GCGTAGGTAGGTAGGTGCGCACCTACCTACCTACGC

Number of atoms	37214
Number of water molecules	11610
Number of ions	106 (Cl, K+)

MOLECULAR DYNAMICS

Force-field(s)	AMBER 99 (bsc0) SPC/E
Solvent	Explicit
Constraints	SHAKE
Electrostatics interactions	PME
Thermostat	Berendsen
Boundary conditions	Periodic
Physical time	1370.0 ns
Time step	0.002 ps

3D structure

Trajectory snapshot with ions and no water

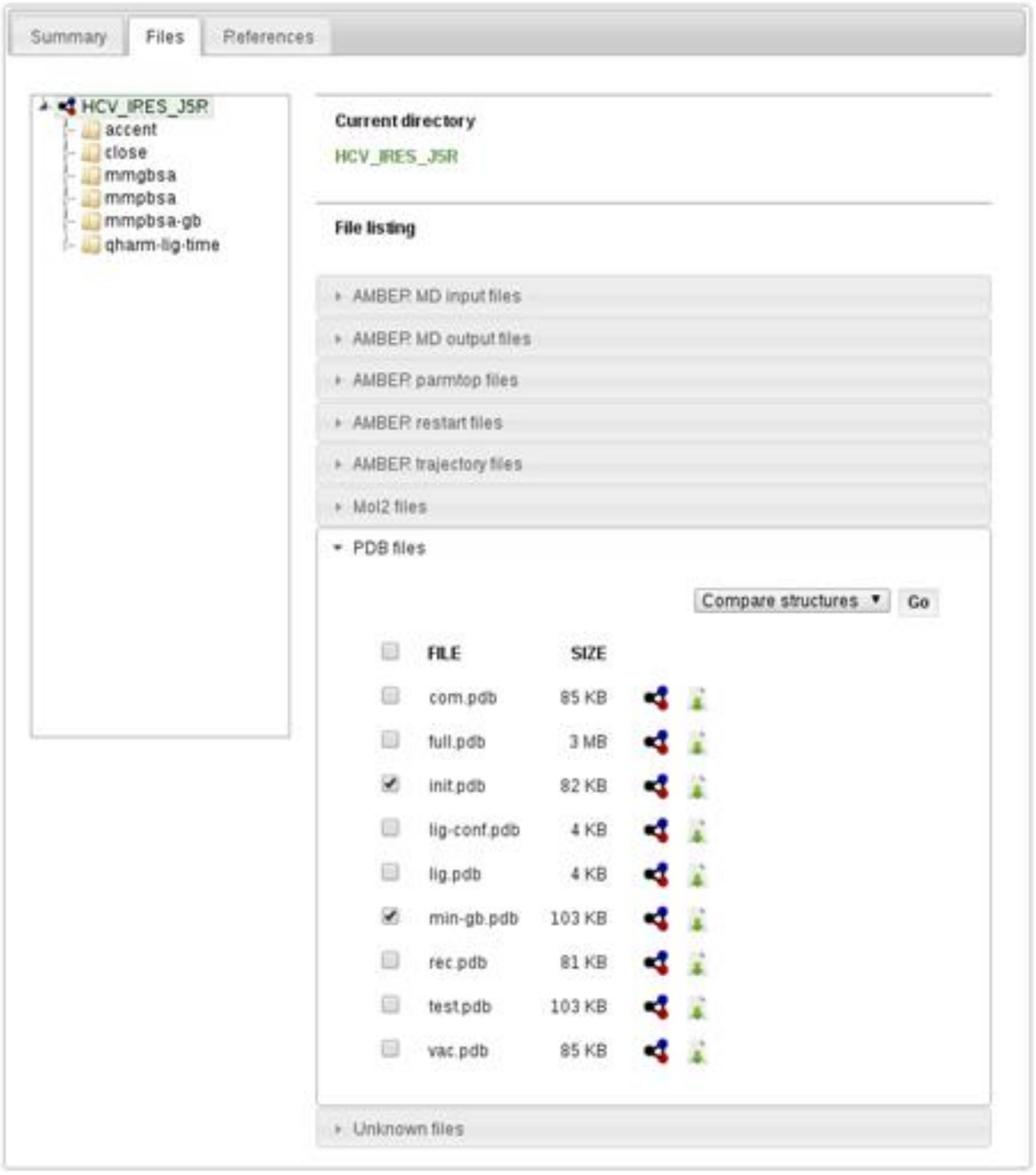
Analysis

Images

Links

Analysis summary

Figure 5.4, iBIOMES web interface: summary page for an MD simulation of DNA including analysis data and a representative 3D structure.



HCV_IPES_J5R
- accent
- close
- mmgbsa
- mmpbsa
- mmpbsa-gb
- qharm-lig-time

Current directory

HCV_IRES_J5R

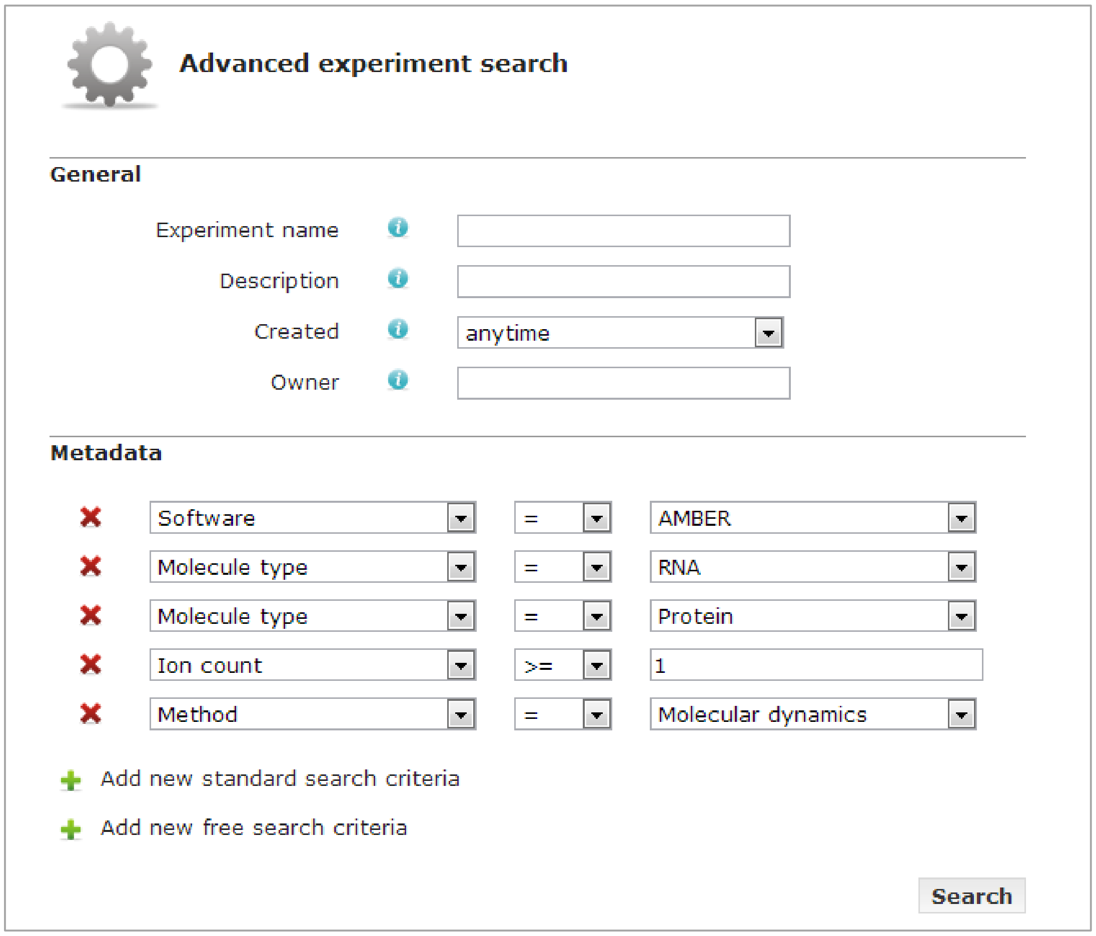
File listing

- AMBER MD input files
- AMBER MD output files
- AMBER parmtop files
- AMBER restart files
- AMBER trajectory files
- Mol2 files
- PDB files

	FILE	SIZE
☐	com.pdb	85 KB
☐	full.pdb	3 MB
✓	init.pdb	82 KB
☐	lig-conf.pdb	4 KB
☐	lig.pdb	4 KB
✓	min-gb.pdb	103 KB
☐	rec.pdb	81 KB
☐	test.pdb	103 KB
☐	vac.pdb	85 KB

- Unknown files

Figure 5.5, iBIOMES web interface: file listing for a particular experiment.



Advanced experiment search

General

Experiment name	
Description	
Created	anytime
Owner	

Metadata

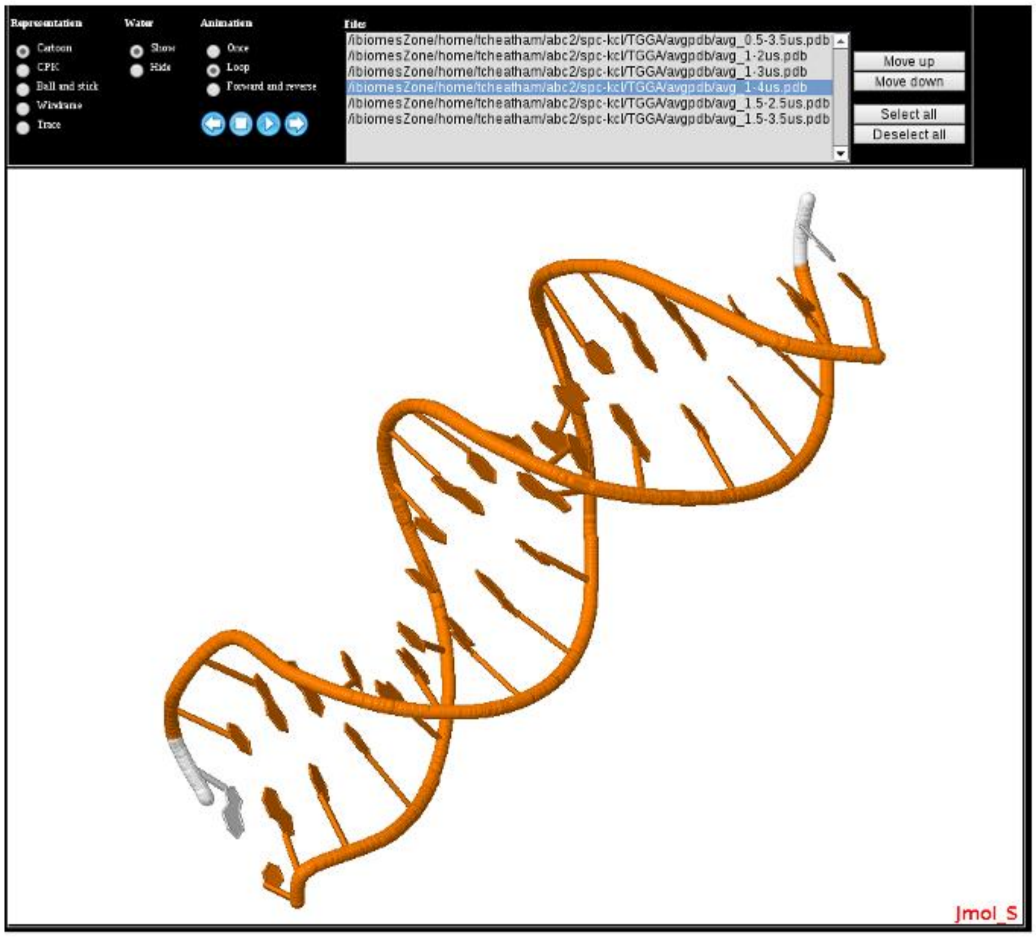
Software	=	AMBER
Molecule type	=	RNA
Molecule type	=	Protein
Ion count	>=	1
Method	=	Molecular dynamics

Add new standard search criteria

Add new free search criteria

Search

Figure 5.6, Advanced experiment search through the web interface. Users can pick metadata attributes and values from the standard catalog or create free-text criteria. This particular example shows how one would search MD simulations of protein/RNA complexes run with AMBER.



Representation: [✓ Cartoon] CPK Ball and stick Wireframe Trace

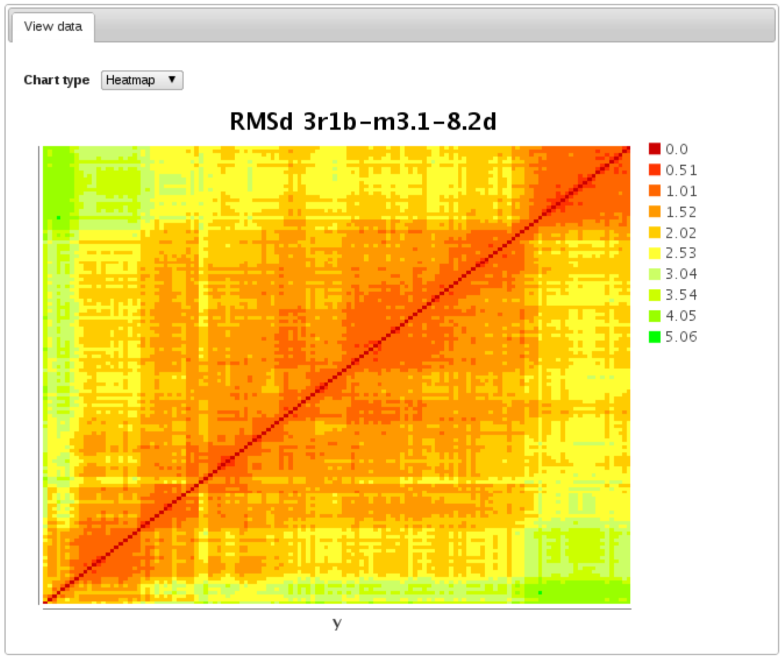
Water: [✓ Show] Hide

Animation: Once [✓ Loop] Forward and reverse

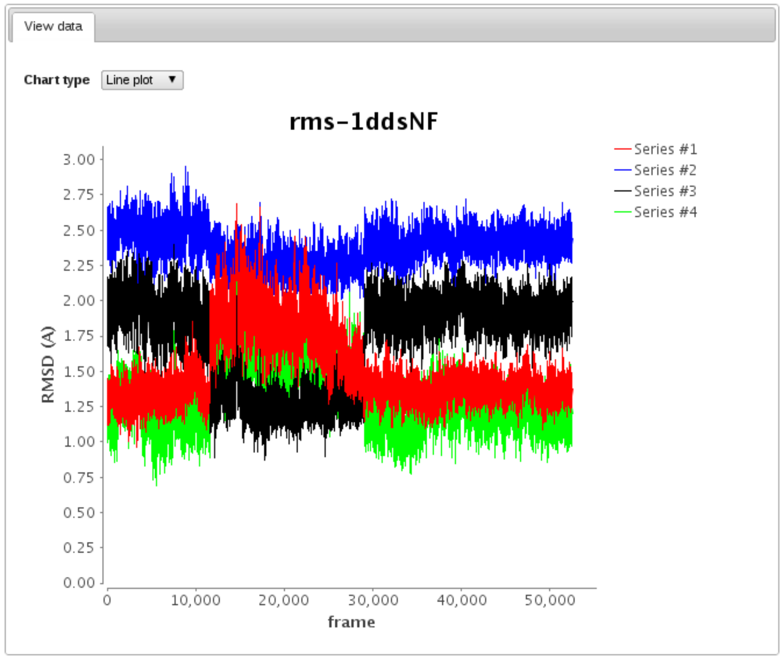
Files

```
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_0.5-3.5us.pdb
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_1-2us.pdb
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_1-3us.pdb
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_1-4us.pdb
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_1.5-2.5us.pdb
/ibiomesZone/home/tcheatham/abc2/spc-kcl/TGGA/avgpdb/avg_1.5-3.5us.pdb
```

Figure 5.7, Integration of Jmol to render and manipulate 3D structures.

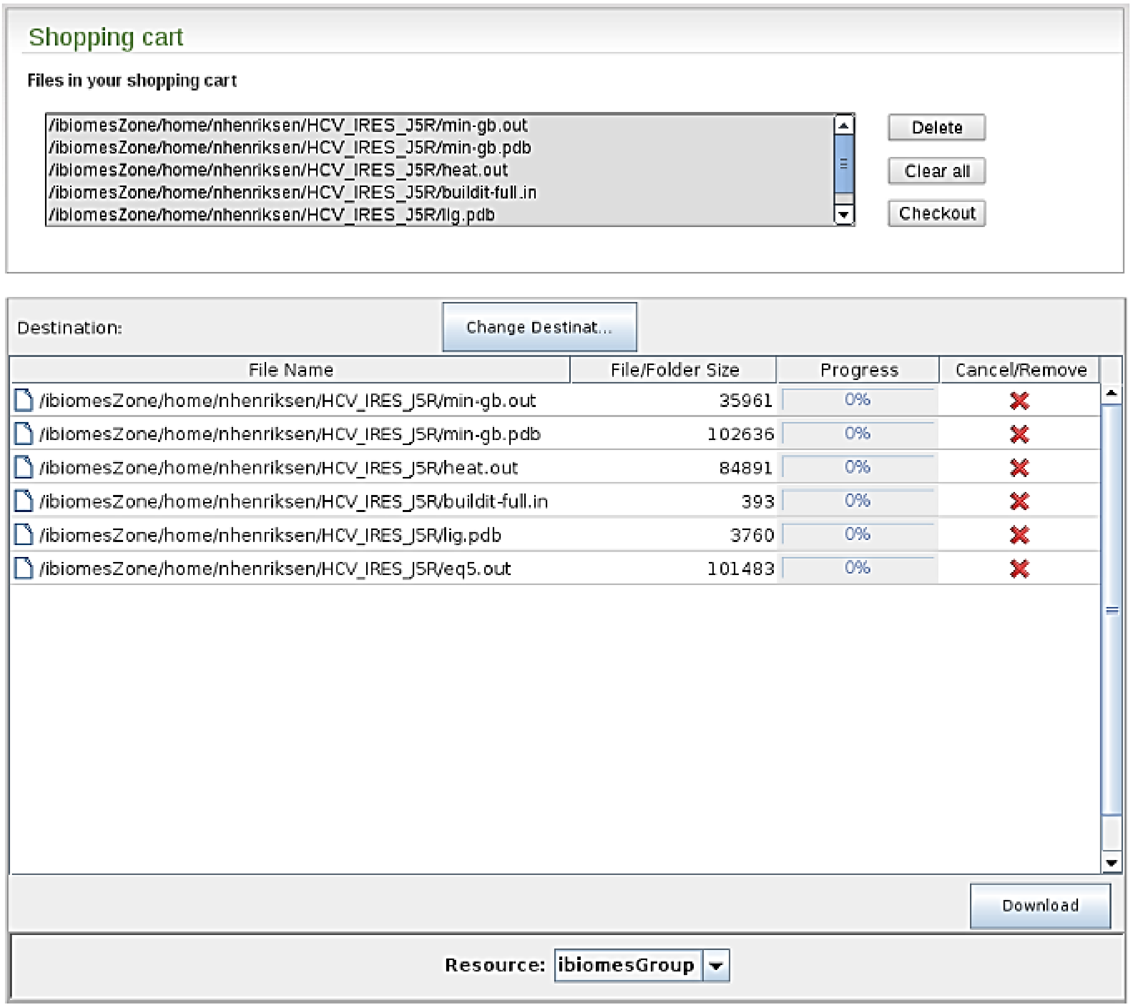


(a)



(b)

Figure 5.8, Plotting tools used in the iBIOMES web interface for data visualization. The plotting service is based on the JFreeChart library and enables comparison of multiple RMSd (root mean square deviation) plots (a) and rendering of RMSd 2D matrices as heatmaps (b).



Files in your shopping cart

```
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/min-gb.out
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/min-gb.pdb
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/heat.out
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/buildit-full.in
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/lig.pdb
```

Destination:

File Name	File/Folder Size	Progress	Cancel/Remove
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/min-gb.out	35961	0%	✗
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/min-gb.pdb	102636	0%	✗
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/heat.out	84891	0%	✗
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/buildit-full.in	393	0%	✗
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/lig.pdb	3760	0%	✗
/ibiomesZone/home/nhenriksen/HCV_IRES_J5R/eq5.out	101483	0%	✗

Resource: ibiomesGroup

Figure 5.9, Integration of the iDrop Lite applet to enable bulk downloads of files through the shopping cart service.

For in-place registration:

```
ibiomes register -i local-dir [-o irods-vpath] [-s software] \
[-x xml-descriptor]
```

For data deposit with transfer:

```
ibiomes push -i local-dir [-o irods-vpath] [-s software] \
 [-x xml-descriptor] [-r default-resc]
```

Arguments:

```
[local-dir] Path to the local directory to parse/register
[irods-vpath] Virtual path to the iRODS collection to be created
[software] Name of the software package used to run the simulation
(e.g., amber, nwchem)
[xml-descriptor] Path to the XML descriptor that specifies metadata
generation rules
[default-resc] Name of the default iRODS resource to use for storage
```

Figure 5.10, iBIOMES commands for in-place registration and standard publication with data transfer

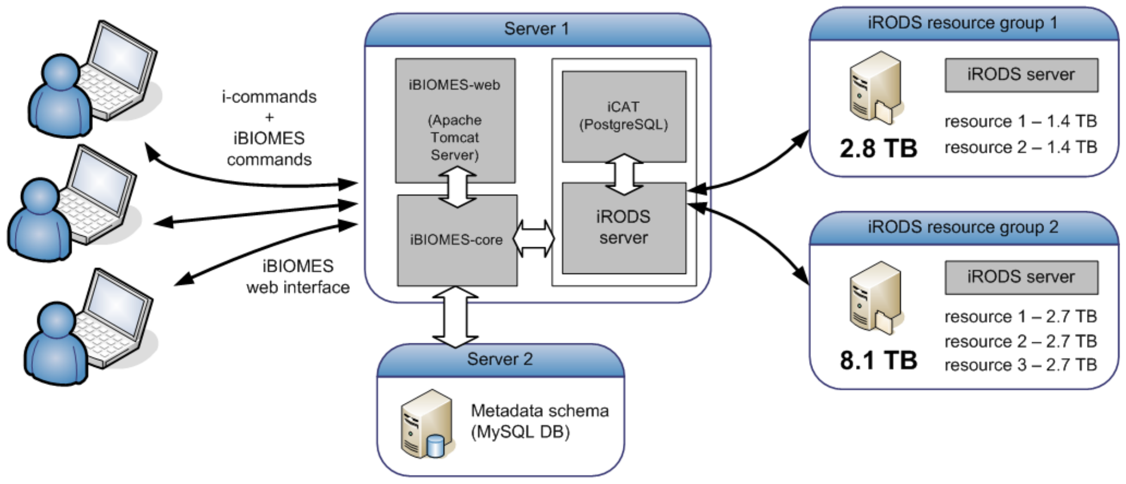


Figure 5.11, Configuration of the iBIOMES infrastructure at the University of Utah (Cheatham lab). Storage resources are distributed over 2 servers and currently offer a 10 TB capacity.

Table 5.1, Simplified view of the iRODS user-metadata table

File ID	Attribute	Value	Unit
1	molecule type	Protein	
1	simulated time	0.5	ms
1	software	AMBER	
2	molecule type	RNA	
2	temperature	300	K

Table 5.2, A subset of the metadata attributes defined in iBIOMES

Category	Attribute	Example values
Molecular System	Water count	*Integer*
	Atom count	*Integer*
	Ion count	*Integer*
	Molecule type	*Protein, RNA, DNA, chemical compound*
	Residue sequence	*ATTCGAAT, ALA PRO HIS LEU, APHL*
	Reference structure	*PDB:1BIV, PubChem:2733526*
Method (general)	General method	*Molecular dynamics, Quantum Mechanics, Coarse-grain Dynamics, QM/MM*
	Boundary conditions	*Periodic, non-periodic*
	Solvent	*Implicit, explicit, in vacuum*
Molecular Dynamics	Force field	*AMBER FF 99, GROMOS 43A1 , ReaxFF*
	Barostat	*Andersen, Berendsen, Parrinello-Rahman*
	Thermostat	*Berendsen, Nose, Nose-Poincare*
	Molecular mechanics integrator	*Verlet, Leapfrog*
	Electrostatics modeling	*Cutoff, Classic ewald, PME, reaction field*
Quantum Mechanics	General QM method	*Hartree-Fock, Moeller-Plesset, DFT, Configuration interaction*
	Level of theory	*SCF, MP2, MP4, CCSD(T)*
	Basis set	*STO-3G, 6-31++G*, cc-pCDVZ*
	Spin multiplicity	*0, 2*
	Total charge	*-1, 0, 1, 2*

References

1. Dror, R. O.; Dirks, R. M.; Grossman, J. P.; Xu, H.; Shaw, D. E., Biomolecular Simulation: a Computational Microscope for Molecular Biology. *Annu. Rev. Biophys.* **2012**, 41, 429-452.

2. Alonso, H.; Bliznyuk, A. A.; Gready, J. E., Combining Docking and Molecular Dynamic Simulations in Drug Design. *Med. Res. Rev.* **2006**, 26, 531-568.

3. Klein, M. L.; Shinoda, W., Large-Scale Molecular Dynamics Simulations of Self-Assembling Systems. *Science* **2008**, 321, 798-800.

4. Shaw, D. E.; Deneroff, M. M.; Dror, R. O.; Kuskin, J. S.; Larson, R. H.; Salmon, J. K.; Young, C.; Batson, B.; Bowers, K. J.; Chao, J. C. Anton, A Special-Purpose Machine for Molecular Dynamics Simulation. In ACM SIGARCH Computer Architecture News, 2007; ACM: 2007; Vol. 35; pp 1-12.

5. Case, D. A.; Cheatham, T. E., 3rd; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M., Jr.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J., The Amber Biomolecular Simulation Programs. *J. Comput. Chem.* **2005**, 26, 1668-1688.

6. Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K., Scalable Molecular Dynamics with NAMD. *J. Comput. Chem.* **2005**, 26, 1781-1802.

7. Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D., CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics Calculations. *J. Comput. Chem.* **1983**, 4, 187-217.

8. Hess, B.; Kutzner, C.; van der Spoel, D.; Lindahl, E., GROMACS 4: Algorithms for Highly Efficient, Load-Balanced, and Scalable Molecular Simulation. *J. Chem. Theory. Comput.* **2008**, 4, 435-447.

9. Plimpton, S., Fast Parallel Algorithms for Short-Range Molecular Dynamics. *J. Comput. Phys.* **1995**, 117, 1-19.

10. Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.;

Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. *Gaussian 09*, Revision C. 01; Gaussian, Inc: Wallingford, CT, 2009.

11. Valiev, M.; Bylaska, E. J.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Van Dam, H. J. J.; Wang, D.; Nieplocha, J.; Apra, E.; Windus, T. L., NWChem: A Comprehensive and Scalable Open-Source Solution for Large Scale Molecular Simulations. *Comput. Phys. Commun.* **2010**, 181, 1477-1489.

12. Schmidt, M. W.; Baldridge, K. K.; Boatz, J. A.; Elbert, S. T.; Gordon, M. S.; Jensen, J. H.; Koseki, S.; Matsunaga, N.; Nguyen, K. A.; Su, S., General Atomic and Molecular Electronic Structure System. *J. Comput. Chem.* **2004**, 14, 1347-1363.

13. Kong, J.; White, C. A.; Krylov, A. I.; Sherrill, D.; Adamson, R. D.; Furlani, T. R.; Lee, M. S.; Lee, A. M.; Gwaltney, S. R.; Adams, T. R., Q-Chem 2.0: A High-Performance Ab Initio Electronic Structure Program Package. *J. Comput. Chem.* **2000**, 21, 1532-1548.

14. *Jaguar*, Version 7.5; Schrödinger, L.L.C.: New York, NY, 2008.

15. *Vienna Ab Initio Simulation Package (VASP)*, Version 5.3.3; 2012.

16. Beveridge, D. L.; Cheatham, T. E., III; Mezei, M., The ABCs of Molecular Dynamics Simulations on B-DNA, Circa 2012. *J. Biosci. (Bangalore, India)* **2012**, 37, 379-397.

17. Lavery, R.; Zakrzewska, K.; Beveridge, D.; Bishop, T. C.; Case, D. A.; Cheatham, T., III; Dixit, S.; Jayaram, B.; Lankas, F.; Laughton, C.; Maddocks, J. H.; Michon, A.; Osman, R.; Orozco, M.; Perez, A.; Singh, T.; Spackova, N.; Sponer, J., A Systematic Molecular Dynamics Study of Nearest-Neighbor Effects on Base Pair and Base Pair Step Conformations and Fluctuations in B-DNA. *Nucleic Acids Res.* **2010**, 38, 299-313.

18. Beveridge, D. L.; Barreiro, G.; Byun, K. S.; Case, D. A.; Cheatham, T. E.; Dixit, S. B.; Giudice, E.; Lankas, F.; Lavery, R.; Maddocks, J. H.; Osman, R.; Seibert, E.; Sklenar, H.; Stoll, G.; Thayer, K. M.; Varnai, P.; Young, M. A., Molecular Dynamics Simulations of the 136 Unique Tetranucleotide Sequences of DNA Oligonucleotides. I. Research Design and Results on d(CpG) Steps. *Biophys. J.* **2004**, 87, 3799-3813.

19. Dixit, S. B.; Beveridge, D. L.; Case, D. A.; Cheatham, T. E.; Giudice, E.; Lankas, F.; Lavery, R.; Maddocks, J. H.; Osman, R.; Sklenar, H.; Thayer, K. M.; Varnai, P., Molecular Dynamics Simulations of the 136 Unique Tetranucleotide Sequences of DNA Oligonucleotides. II: Sequence Context Effects on the Dynamical Structures of the 10 Unique Dinucleotide Steps. *Biophys. J.* **2005**, 89, 3721-3740.

20. *The Grid 2: Blueprint for a New Computing Infrastructure*. second ed.; Morgan Kaufmann: San Francisco, CA, 2003.

21. Ng, M. H.; Johnston, S.; Wu, B.; Murdock, S. E.; Tai, K.; Fangohr, H.; Cox, S. J.; Essex, J. W.; Sansom, M. S. P.; Jeffreys, P., BioSimGrid: Grid-Enabled Biomolecular Simulation Data Storage and Analysis. *Future Gener. Comp. Sy.* **2006**, 22, 657-664.

22. Simms, A. M.; Toofanny, R. D.; Kehl, C.; Benson, N. C.; Daggett, V., Dynameomics: Design of a Computational Lab Workflow and Scientific Data Repository for Protein Simulations. *Protein Eng. Des. Sel.* **2008**, 21, 369-377.

23. Alfredsson, M. eMinerals: Science Outcomes Enabled by New Grid Tools. In Proc. UK eScience All Hands Meeting, 2005; 2005; pp 788-795.

24. Calleja, M.; Bruin, R.; Tucker, M. G.; Dove, M. T.; Tyer, R.; Blanshard, L.; Van Dam, K. K.; Allan, R. J.; Chapman, C.; Emmerich, W., Collaborative Grid Infrastructure for Molecular Simulations: The eMinerals Minigrid as a Prototype Integrated Compute and Data Grid. *Molecular Simulation* **2005**, 31, 303-313.

25. Meyer, T.; D'Abramo, M.; Hospital, A.; Rueda, M.; Ferrer-Costa, C.; Perez, A.; Carrillo, O.; Camps, J.; Fenollosa, C.; Repchevsky, D.; Lluis Gelpi, J.; Orozco, M., MoDEL (Molecular Dynamics Extended Library): A Database of Atomistic Molecular Dynamics Trajectories. *Structure* **2010**, 18, 1399-1409.

26. Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Lluis Gelpi, J., MDWeb and MDMoby: An Integrated Web-Based Platform for Molecular Dynamics Simulations. *Bioinformatics* **2012**, 28, 1278-1279.

27. Vohra, S.; Hall, B. A.; Holdbrook, D. A.; Khalid, S.; Biggin, P. C., Bookshelf: A Simple Curation System for the Storage of Biomolecular Simulation Data. *Database: the Journal of Biological Databases and Curation* **2010**.

28. Rajasekar, A.; Moore, R.; Hou, C.; Lee, C. A.; Marciano, R.; de Torcy, A.; Wan, M.; Schroeder, W.; Chen, S. Y.; Gilbert, L., iRODS Primer: Integrated Rule-Oriented Data System. *Synthesis Lectures on Information Concepts, Retrieval, and Services* **2010**, 2, 1-143.

29. Fielding, R. T., Chapter 5: Representational State Transfer (REST). *Architectural Styles and the Design of Network-based Software Architectures, Dissertation* **2000**.

30. Baru, C.; Moore, R.; Rajasekar, A.; Wan, M. The SDSC Storage Resource Broker. In Proceedings of the 1998 Conference of the Centre for Advanced Studies on Collaborative Research, 1998; IBM Press: 1998; p 5.

31. Chiang, G.-T.; Clapham, P.; Qi, G.; Sale, K.; Coates, G., Implementing a Genomic Data Management System Using iRODS in the Wellcome Trust Sanger Institute. *BMC Bioinf.* **2011**, 12, 361.

32. Goff, S. A.; Vaughn, M.; McKay, S.; Lyons, E.; Stapleton, A. E.; Gessler, D.; Matasci, N.; Wang, L.; Hanlon, M.; Lenards, A.; Muir, A.; Merchant, N.; Lowry, S.; Mock, S.; Helmke, M.; Kubach, A.; Narro, M.; Hopkins, N.; Micklos, D.; Hilgert, U.; Gonzales, M.; Jordan, C.; Skidmore, E.; Dooley, R.; Cazes, J.; McLay, R.; Lu, Z.; Pasternak, S.; Koesterke, L.; Piel, W. H.; Grene, R.; Noutsos, C.; Gendler, K.; Feng, X.; Tang, C.; Lent, M.; Kim, S.-J.; Kvilekval, K.; Manjunath, B. S.; Tannen, V.; Stamatakis, A.; Sanderson, M.; Welch, S. M.; Cranston, K. A.; Soltis, P.; Soltis, D.; O'Meara, B.; Ane, C.; Brutnell, T.; Kleibenstein, D. J.; White, J. W.; Leebens-Mack, J.; Donoghue, M. J.; Spalding, E. P.; Vision, T. J.; Myers, C. R.; Lowenthal, D.; Enquist, B. J.; Boyle, B.; Akoglu, A.; Andrews, G.; Ram, S.; Ware, D.; Stein, L.; Stanzione, D., The iPlant Collaborative: Cyberinfrastructure for Plant Biology. *Front. Plant. Sci.* **2011**, 2.

33. Herráez, A., Biomolecules in the Computer: Jmol to the Rescue. *Biochem. Mol. Biol. Educ.* **2006**, 34, 255-261.

34. Bernstein, F. C.; Koetzle, T. F.; Williams, G. J. B.; Meyer, E. F.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; Tasumi, M., The Protein Data Bank. *Eur. J. Biochem.* **2008**, 80, 319-324.

35. Wang, Y.; Xiao, J.; Suzek, T. O.; Zhang, J.; Wang, J.; Bryant, S. H., PubChem: A Public Information System for Analyzing Bioactivities of Small Molecules. *Nucleic Acids Res.* **2009**, 37, W623-W633.

36. Allen, F. H.; Taylor, R., Research Applications of the Cambridge Structural Database (CSD). *Chem. Soc. Rev.* **2004**, 33, 463-475.

37. Shahrokh, K.; Orendt, A.; Yost, G. S.; Cheatham, T. E., 3rd, Quantum Mechanically Derived AMBER-Compatible Heme Parameters for Various States of the Cytochrome P450 Catalytic Cycle. *J. Comput. Chem.* **2012**, 33, 119–133.

CHAPTER 6

IBIOMES LITE: SUMMARIZING BIOMOLECULAR SIMULATION DATA IN LIMITED SETTINGS[1]

Abstract

As the amount of data generated by biomolecular simulations dramatically increases, new tools need to be developed to manage these data at the individual investigator or small research group level. In this paper we introduce iBIOMES Lite, a light-weight tool for biomolecular simulation data indexing and summarization. The main goal of iBIOMES Lite is to provide a simple interface to summarize computational experiments in a setting where the user might have limited privileges and limited access to IT resources. A command-line interface allows the user to summarize, publish, and search local simulation datasets. Published datasets are accessible via static HTML pages summarizing the simulation protocol and presenting analysis data graphically. The publication process is customized via XML descriptors while the HTML summary template is customized though XSL stylesheets. iBIOMES Lite was tested on different platforms and at several national computing centers against various datasets generated through classical and quantum molecular dynamics, quantum chemistry, and QM/MM.

[1] Reproduced in part with permission from Thibault, J.C., Cheatham III, T.E., and Facelli, J.C. (2014). iBIOMES Lite: Summarizing Biomolecular Simulation Data in Limited Settings. Journal of Chemical Information and Modeling, 54 (6), 1810-1819. Copyright 2014 American Chemical Society.

The associated parsers currently support AMBER, GROMACS, Gaussian, and NWChem dataset publication. The code is available at: https://github.com/jcvthibault/ibiomes.

Background

The use of high-performance computing resources to push the limits of biomolecular simulations has been a necessity for decades. As more computational power becomes available, researchers can tackle larger systems and longer time scales. While it was common practice to run the simulations on remote clusters and bring back the resulting data to the home institution, this paradigm now breaks down. Data have to be postprocessed directly at the source to minimize data movements and minimize the amount of disk space necessary for storage. For example trajectories can be compressed and/or stripped of unnecessary information (e.g., solvent) before being copied over. Another approach is to simply run the analysis remotely, where the data reside. No matter which approach is preferred, researchers need to deal with huge amount of data distributed over local and national resources.

Several repository architectures have been proposed to manage large biomolecular simulation datasets in a distributed environment. BioSimGrid[1] was deployed in the UK to integrate several computational centers into a grid, where data could be deposited, searched and analyzed. Trajectory and provenance metadata were stored in a relational database. iBIOMES[2] on the other hand offers a distributed infrastructure that allows biomolecular simulation data indexing with data deposit (explicit copy) or in-place registration to avoid data movements. Trajectory files are stored and indexed via the iRODS distributed file system,[3] where metadata are represented as Attribute-Value-Unit triplets. While these approaches might work well to manage large distributed environments, the deployment of such infrastructure depends on access to substantial IT expertise and resources, such as web servers, relational databases, and distributed file systems, which may not be available to many single investigators or small research groups. Many researchers also depend on local or national computational and storage resources that are allocated for a finite period of time. Usage of these resources is usually very restrictive for security reasons and the installation of heavy components such as databases is not an option to manage the data hosted at these remote locations. Another limitation of current repositories is the need to copy the simulation data to a remote server for publication. This can be a tedious task that requires extra storage cost if a copy of the data has to be kept at its original location. In this paper we introduce iBIOMES Lite, a new tool for biomolecular simulation data indexing and summarization, designed to run in limited settings, where the users might have limited privileges and limited access to IT resources. A command-line interface allows the user to summarize, publish, and search simulation datasets locally or remotely via secure shell (SSH). Published datasets are summarized through a static web interface that describes the simulation protocols and graphically represent analysis results. iBOMES Lite can be easily installed on any data server to enable summarizations of old datasets and figure out what their content is and what methods were used, or to facilitate progress tracking by exposing current simulation results. In contrast with simple tools such as Bookshelf[4] and UMM-MoDEL[5] that have been proposed to publish simulation data, but exhibit dependencies on database components, iBIOMES Lite allows data indexing and summarization while removing dependencies on external components that would require root access or special support for deployment.

<u>Design</u>

Scope and requirements

iBIOMES Lite’s goal is to provide the means for researchers to index and summarize simulation data in limited settings, so they can keep track of their lab work and share progress or results with collaborators. The main user action supported by iBIOMES Lite is the publication of experiments: the user specifies a file directory or subdirectory that contains all the simulation files (input and output data), then with minimal input from the user, the tool generates a detailed description of the computational experiment workflow along with textual and graphical summaries, rendered through a simple web interface. Once an experiment is published it can be searched via keywords representing the experiment metadata (e.g., molecule name, residue sequence, computational method). Unlike the full fledge iBIOMES repository,[2] iBIOMES Lite does not provide access to the files associated to the published experiments. All files are categorized and listed, but only files presenting analysis data are made available for download. This limitation was required to keep simplicity as a key design criterion for this tool. This criterion was applied at 3 different levels: deployment, usage, and customization as follows:

- <u>Deployment</u>: the tool should be able to run in most environments, independently from the operating system running on the host (e.g., Unix, Windows). The tool should also be able to run whether a graphical user interface is available or not. Root permissions should not be a prerequisite to install the program. This can be achieved by removing dependencies on heavy-weight components such as databases, web servers, or specific file systems.

- Usage: the tool should be usable in a multiuser and distributed environment by providing simple commands. The command-line interface provides a Unix-like interface to summarize simulation data, publish them into a static HTML web site, and perform keyword searches.
- Customization: the publication process should be easily customizable by the user so that the resulting summaries provide an accurate and pertinent representation of the raw data. The actual code should not have to be modified to perform such customization. Instead customization should be enabled through templates, and configuration files.

Web interface

The entry point for the web interface is a page listing all the published experiments, as shown in the iBIOMES Lite demonstration instance presented in Figure 6.1. General information about the experiments (e.g., method, targeted molecular system, software package) is provided and can be used to sort the listing. By selecting one of the listed experiments the user can access more details. Currently, each experiment is associated to 4 different HTML pages. The summary page (Figure 6.2) presents a summary of the experiment protocol along with possible analysis data, plots and 3D structures, rendered via Jmol.[6] A second HTML page provides a tree view of the protocol used in the experiment, so that the user can access the details of interest, while keeping the overall picture of the workflow (Figure 6.3). A third HTML page provides a tree view that allows the user to browse the directory and subdirectories associated to the experiment and list their content (Figure 6.4).

Finally a last HTML page gives details about the execution of the tasks and the computing environment (Figure 6.5). Execution times and resources used to run the tasks (e.g., number of CPUs and GPUs) are reported, along with hardware information (e.g., GPU architecture). Tasks that did not terminate correctly are flagged. This view is intended for users to track the progress of current simulations and assess the performance of their simulation engine within the host environment.

Implementation

Overview

iBIOMES Lite was implemented in Java 7 to ease the development of a platform-independent tool. Although Java 6 is arguably a more popular version, Java 7 offers enhanced file I/O libraries (NIO 2) that might prove to be useful for future developments (e.g., file change listeners, file tree searches), and it is still available at most US computing centers. A set of Bash scripts for Unix-like operating systems (i.e., Linux and Mac OS-X) and Win32 (.bat) scripts for Windows were written to wrap the Java calls into simple commands. These scripts can be easily called in a console locally or remotely, via SSH for example.

Publication process

Users publish computational experiments to iBIOMES Lite to create HTML summaries and index their data for searches. A user publishes a computational experiment by specifying a directory or subdirectory that contains all the simulation files (input and output) and the name of the software package that was used to generate these files (Figure 6.6). A set of file parsers extract topology, method, and parameter information to generate a representation of the simulation workflow, based on the data model introduced in previous work.[7] The workflow and file tree structures are stored as XML files then transformed into several HTML pages via XSL (eXtensible Stylesheet Language[8]). Plots are generated for analysis files when applicable then stored in the iBIOMES Lite web directory along with the HTML files.

The final output of the publication process is a set of XML files, static HTML files, images, and other analysis data files (e.g., spreadsheets). These output files can be exposed via an HTTP server such as Apache (http://httpd.apache.org/), or viewed locally if a graphical user interface is available. If neither option is available, the files can also be copied to a different host for rendering. Since the HTML is not generated on-the-fly by server-side code the web content can always be copied without information loss.

In the next sections we describe in more details the data extraction step performed by the file parsers and the data transformation step used to generate the HTML summaries.

Parsers

The role of the parsers is to map a given computational experiment file tree on disk to a logical representation of the protocol and output of the experiment. The data model introduced in [7] was used to guide the logical representation, for both the definition of the Java classes and the XML schema used to represent individual computational experiments, i.e., the simulations. The parsers work at the file level, extracting important data or metadata for file summary, and at the file tree level, trying to build the logical model based on the file directory structure and the file-extracted data.

File parsers

The file parsers are format-specific, although they are expected to build certain common objects based on their type: topology, parameter/method, or hybrid. For example both the AMBER parameter/topology and Protein Structure File (PSF) parsers are expected to build an object representing a molecular system, composed of one or multiple molecules, each represented by residues and/or atoms. On the other hand the AMBER MD input and NAMD configuration file parsers are building objects representing the methods and parameters used to run a computational task. Implementation of the parsers then requires understanding of the target format and the expected object(s) to build. All parsers target the data model introduced in[7] to provide a common representation of the computational protocol that is not software-specific. The list of current parsers provides different levels of support for various software packages, including AMBER,[9] GROMACS,[10] NAMD,[11] NWChem,[12] and Gaussian.[13]

File tree parsers

The implementation of file tree parsers is not as straight forward. The structure of a file is inferred from its format while the structure of a directory does not follow any strict rule. While we cannot force users to store their files following a given directory structure, manual inspection of files structure from many computational experiments performed in our lab by numerous graduate students and post docs lead us to assume that the protocol of the computational experiment can be inferred by parsing certain files if the original owner can provide a description of the file tree structure and the naming conventions used to organize the data.

The preprocessing step in the mapping process is to parse all the files in the input directory and its subdirectories using the file-specific parsers. The resulting file tree associates each file with a set of descriptive data about the molecular system or computational methods. The second step is to build a logical representation of the computational experiment protocol using these objects. When publishing a new experiment the user needs to specify the main software package that was used to run the simulations (e.g., AMBER, NAMD, Gaussian, NWChem). Depending on this argument different rules are used to build the logical representation of the experiment. For example in AMBER, both MD input and MD output files can be used to retrieve the methods and parameters of a run. As for most software packages the output/log files are preferred over input files to extract this type of data. Output files are typically richer as they usually repeat information from the input file(s) and provide explicit values to parameters that have not been set in the input, but which are used as the default values in the particular software. Output files can also present some calculation details, such as the evolution of the energy of system over a certain cycle of iterations, that can be easily exposed and of potential value to better understand the experiment protocol.

Other rules can be triggered based on the computational method used or the type of calculation performed. For example if minimization tasks and MD tasks are detected within the experiment, minimization tasks are grouped together, while MD tasks are divided into a “heating” process, an “equilibration” process and a “production MD” process. Heating tasks represent MD runs where temperature of the system is slowly increased, to eventually reach a reference temperature for the production runs. Distinction between equilibration and production runs is currently made based on the textual description of the task if it is available. Regular expressions were created to detect keywords such as “production,” “prod,” “equilibration,” and “equil.”

For Replica-Exchange MD (REMD), some extra step might be needed to group replicas for the same run together. In AMBER for example, an output file is created for each replica. In our data model, all replicas for a single run are grouped together under a single REMD task instead of having separate MD tasks representing individual replicas. Each REMD task is described like any other MD task and it also has a certain number of replicas and a type of exchange (e.g., temperature, Hamiltonian, multidimensional). This representation helps summarizing the data, especially when running REMD simulations with hundreds of replicas. By default REMD output files stored in the same folder are assumed to represent replicas from the same group. This would apply for example if a user stored 3 4-replica REMD runs in 3 different folders with each 4 output files. Experience shows that this approach is not unique, and some people might prefer to have all REMD output in a single folder. Replica identification and grouping is then based on file naming conventions. Using the same example, a user could store all the REMD output files in a single folder and name the files using the pattern that identifies both the run and the replica within this run, such as:

$$\texttt{remd.[ID}_{\texttt{RUN}}\texttt{].[ID}_{\texttt{REPLICA}}\texttt{].out,}$$

$$\texttt{where } 0 \leq \texttt{ID}_{\texttt{RUN}} \leq 2 \texttt{ and } 0 \leq \texttt{ID}_{\texttt{REPLICA}} \leq 3.$$

The user can specify this type of naming convention in the iBIOMES Lite general configuration file or at run time using the *–remd* command line argument. If no run identifier is present in the name pattern then grouping is solely based on the directory structure. This type of rule-based grouping is currently applied to REMD tasks only but it could be expended to include any type of parallel enhanced sampling task.

Data transformations

XML representation

After the logical model of an experiment is built within the Java code it is stored on disk as an XML file. Mapping between the Java object-oriented data model and the XML schema is performed via JAXB (Java Architecture for XML Binding). An example of such XML is presented in Appendix E. A second XML file is generated based on the file tree structure, where each file is associated to a set of metadata, represented as attribute-value-units (AVU) triplets. This representation is very similar to the approach used for the iBIOMES repository[2] to enable indexing within iRODS (Integrated Rule-Oriented Data System[3]). An example of such an XML file tree is illustrated in Appendix E. The AVUs are derived from the objects extracted by the file parsers, such as molecular system definitions or parameter sets. Each of these entities implement a *getMetadata()* method that translates the logical entity (object) into a list of AVUs. For example the *getMetadata()* method for the *Thermostat* class will generate AVUs for the followings attributes: THERMOSTAT_ALGORITHM (e.g., Berendsen, Langevin) and THERMOSTAT_TIME_CONSTANT if applicable.

These XML documents provide two different perspectives on the data: one that emphasizes on the experiment protocol, the logical view, and another one that emphasizes on the physical organization of the input and output files. While the first view can provide some insight on the protocol used to run the simulations, the second view enables simple data indexing via keywords. A copy of these XML files is stored directly in the experiment folder. Another copy is pushed to the iBIOMES Lite web folder, in a subdirectory dedicated to the experiment. A separate XML document representing the list of published experiments is also updated by copying experiment-level AVUs from the XML document storing the experiment file tree.

Analysis data

Beside the experiment protocol and the file tree, iBIOMES Lite can present analysis data in the experiment summary page. The user can edit an XML configuration file to define which piece of data should be presented and how it should be presented. This is achieved by associating file name patterns to analysis descriptions, as introduced in iBIOMES.[2] Any file that is marked as analysis data is copied to the iBIOMES Lite web folder to enable display and/or download. For example PDB files that are marked as analysis data can be rendered via Jmol,[6] and image files (e.g., PNG, JPEG) are presented as thumbnails linking to a copy of the original picture. For column delimited text files (e.g., tab- or comma-delimited files) the tool attempts to create a graphical representation of the content. The XML configuration files can be used to define the type of plot to be generated (e.g., line plot, histogram, heatmap), its labels, units, and title. The resulting plot is exported as an image and copied over to the iBIOMES Lite web folder, along with the original data file.

Transformation

Once the XML files and data files have been copied to the iBIOMES Lite web directory, all data and metadata of interest are ready to be visually rendered by transforming the XML into HTML. Multiple XSL stylesheets define the mappings between the XML and the various HTML pages necessary to list the published experiments and provide details about individual experiments. The actual XSL 2.0 based transformation process in the Java code is performed via the Saxon processor.[14] Since XSL stylesheets are defined as separate documents one could easily customize these HTML templates to fit their need.

Shared iBIOMES Lite web folder for multiuser use

iBIOMES Lite allows multiple users to share the same web directory to publish experiments. This means that all the members of a lab for example can publish experiments stored on a shared file system to a single portal. From a user-interface perspective, information about the publication event needs to be tracked: each experiment is associated to a publication date (different from the dataset creation date) and a publisher (i.e., the file system username). From a publication perspective, safeguards have to be created to ensure data integrity when two users try to publish an experiment simultaneously. If both users try to publish the same experiment then one should be blocked to allow the other user's action to parse the associated directory and generate the descriptor files. Whether the target experiments are different or not, the web directory containing the listing and the index of experiments should not be updated concurrently.

A locking system was implemented to prevent concurrent updates. If somehow two users are trying to publish the same experiment folder concurrently, the second user's publication action is automatically cancelled and the user is warned. If two users are trying to publish different experiments simultaneously, updates from the second user on the experiment listing will be queued until the first users' publication process is over.

Commands

Various Unix-like commands are available to manage the published experiments in iBIOMES Lite. A complete description of these commands is available on the iBIOMES Wiki (http://ibiomes.chpc.utah.edu/mediawiki/). Here we only present a summary of the most important ones: the publish (`ibiomes-lite-publish`), the search (`ibiomes-lite-search`), and clean (`ibiomes-lite-clean`) commands.

Publish experiments

To publish an experiment into iBIOMES Lite – i.e., to parse the experiment folder and generate the associated web content – one should use the *ibiomes-lite-publish* command:

```
ibiomes-lite-publish -i <experiment-dir> [-s software] [-x xml-descriptor] [...]
[experiment-dir] Path to the root of the experiment directory
[software] Name of the software package used to run the simulation/calculations (e.g., amber, nwchem)
[xml-descriptor] Path to the XML descriptor that specifies metadata generation rules. If no file is specified default values defined in the API are used.
```

Search experiments

iBIOMES Lite offers a simple search function: the user provides a list of keywords that are matched against the AVU values in the XML document listing all the published experiments. Paths to experiments that contain all provided keywords are returned. Searches are performed via the *ibiomes-lite-search* command, defined as:

```
ibiomes-lite-search < keywords >
[keywords] List of keywords separated by '+' character. Wildcards can
be specified using '%'. Example:
ibiomes-lite-search %dynamics+rna+amber.
2 experiment(s) found:
      [0] /home/user1/ibiomes/test/amber/rnamodrd
      [1] /home/user1/ibiomes/test/amber/tutorial1
```

Clean web content

Remove content (XML and HTML) from iBIOMES Lite website. XML descriptors at the experiment directory level are conserved, and can be published again. If the -i option is not specified then all experiments are removed:

```
ibiomes-lite-clean
ibiomes-lite-clean -i < experiment-dir >
[experiment-dir] Physical path to the experiment to remove from iBIOMES
Lite.
```

Tests in limited settings

Methods

A critical test for iBIOMES Lite is to demonstrate its ability to work in a variety of environments, including large computational clusters hosted by national centers and single PI labs. A successful deployment here is defined by the following criteria:

1. All prerequisites (i.e., Java 7) are installed or can be installed on the targeted system
2. The user can install iBIOMES Lite on the targeted system, i.e., copy the files and set up the necessary environment variables, and configuration parameters.

3. The user can publish datasets within the targeted system and visualize the generated website within this system or an external one (e.g., home institution).
4. To demonstrate these capabilities iBIOMES Lite was deployed on various machines, such as desktop computers and laptops running different operating systems, and at various US computational centers.

Results

iBIOMES Lite was successfully deployed on different desktop computers and laptops, running the following operating systems: Linux (Fedora Core 18), Windows 7, and Mac OS X 10. iBIOMES Lite was also deployed at the following facilities: the Center for High Performance Computing (CHPC) at the University of Utah, the National Center for Supercomputing Applications (NCSA), the Texas Advanced Computing Center (TACC), and the San Diego Supercomputing Center (SDC). The actual computational environments targeted for testing purpose are described in Table 6.1.

More detailed benchmarking on the parser was performed on Blue Waters (NCSA) and Stampede (TACC). The dataset descriptions and associated directory parsing timings are reported in Table 6.2. All the reported timings were obtained by submitting several batch jobs to these two clusters, using a single computational node. The reported average and standard deviation (Std. dev.) for the processing times were calculated based on 10 jobs for each dataset.

Dependence between log file (AMBER MD output) sizes and parser execution times is presented in Figure 6.7. As expected, the larger the aggregated size of all log files the longer the execution time since MD output files are the main target of the parsers. The timings presented here are only presented as a rough estimate for various types of AMBER datasets. In our example datasets the number of topology files (e.g., PDB, AMBER parameter/topology) is fairly small compared to the number of MD output files but the timings are still dependent on these files. For example if a large number of PDB files representing trajectory snapshots or representative structures with solvent information are present in the input directory, the MD output might not have as much impact on the overall parsers' performance. Note that trajectory files (e.g., AMBER NetCDF, CHARMM DCD) are not actually parsed since they are typically very large (~MB-TB) and they do not provide extra information about the topology or methods used in the simulation.

The parsers were also tested on Blue Waters using an interactive session. The parsers seem to be faster with an average execution time of 94.20 seconds, versus 119.4 seconds for the equivalent batch job. The standard deviation was higher (14.85 seconds vs. 2.1 seconds), which can be explained by the fact that the interactive node was shared with other users running various tasks.

Discussion

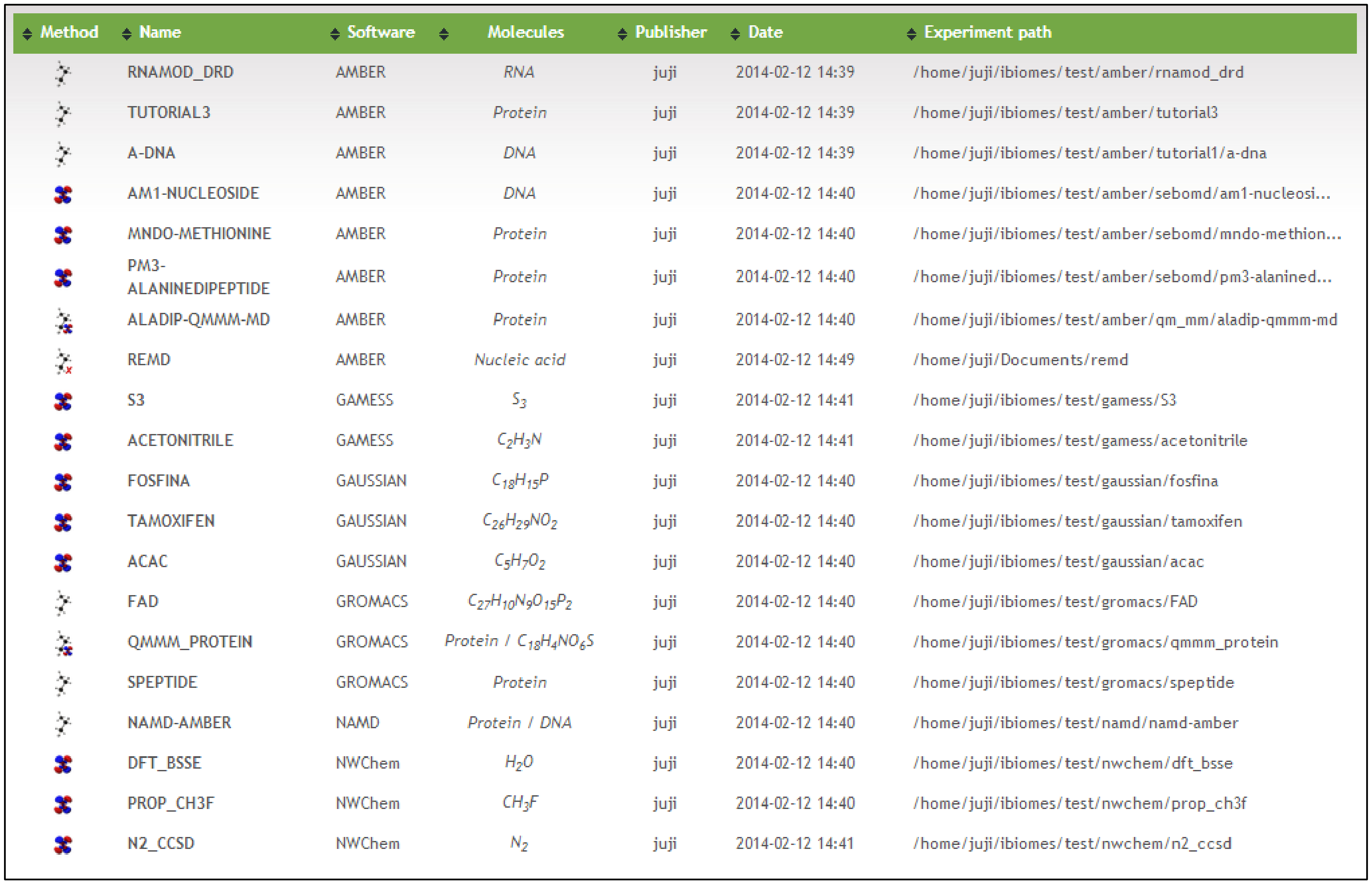
Thanks to its simplicity, iBIOMES Lite can be deployed in limited environments where users have limited permissions and no access to heavy components such as database system managers. More importantly, we showed here that iBIOMES Lite can be used at major computational centers where Big Data is generated. Our current parsers and protocol model builders may not be adapted to all types of directory structure, but this limitation should be circumvented in the future by including more configurable rules based on naming conventions, file content, computational methods, and textual descriptions to enable an accurate representation of the experiment protocol with minimal input from the user.

Summarization does not require bringing back the raw data to the home institution: iBIOMES Lite can be run at the source despite the limitations due to security concerns in such infrastructures. Since the published summaries are static and provide a compressed view of the simulation, the results of the publications can be easily copied to a new location for rendering via the web, or simply to centralize the summaries from different computing centers at a single location. Scripts could be created to automate this process, as well as to regenerate the summaries to make sure that they are up to date with the associated raw data. Since the publication process is performed via a command line interface, the iBIOMES Lite summarization step can be added to a regular simulation job description when running in a cluster. Another alternative when targeting data hosted at a computational center is to run the publication process via an interactive session. For very large datasets with thousands of files the parsers might take over half an hour to go through all the files. Running such tasks on the login nodes of a cluster is usually not recommended by the hosting institution as other users might observe a dramatic slowdown when trying to access their data or submit a job. Most computing centers allow users to request interactive sessions, which are usually provided within minutes, unlike batch job submissions which might stay queued for hours or days.

Although most demonstrations for iBIOMES Lite have been done through the publication of AMBER-generated datasets, the parsers support datasets generated by other MD engines such as GROMACS and NAMD. The development of the data model and parsers has been guided by our experience with AMBER but the support for other software packages has allowed us to avoid software-specific data representations and parsing rules. Parsers for QM datasets (e.g., GAUSSIAN, NWChem) were also developed to demonstrate the generalizability of the data model and the web interface. Although nowadays MD is a de facto standard approach to run biomolecular simulations, QM cannot be excluded from this realm. First MD can be dependent on QM when new force field parameters have to be created for nonstandard residues or small ligands. Then QM has promise in the study of biomolecules, at least for small systems.[15] The inclusion of less common and more complex methods in the data model such as Replica-Exchange MD, QM/MM and Quantum MD has proven the decomposition of parameters into sets of method-specific parameters to be fairly generalizable. These methods are currently supported only for the AMBER software package, which enables QM/MM MD,[16] Semi-empirical Born-Oppenheimer MD (SEBOMD[17]), and replica-exchange MD. The initial rationale behind the development of iBIOMES Lite was the need for a simple tool that would be able to mimic the features offered by the iBIOMES repository[2] in a non-distributed environment controlled by a strict security policy. This has been a successful attempt as iBIOMES Lite can create rich summaries with graphical rendering (Jmol, plots) and basic search capabilities. One advantage of iBIOMES Lite over the distributed repository is the ability to provide a detailed and logical description of the computational experiment protocol via XML transformation. The current AVU model used by the iBIOMES repository to index data is very flexible but relationships between data elements cannot be described. The addition of a relational database to the repository architecture to keep track of the experiment workflow is part of our effort to provide a generic infrastructure for biomolecular simulation data sharing.[7] One of the major limitations of iBIOMES Lite, by design, is the fact that the web interface does not provide access to the raw data. iBIOMES Lite is not a replacement for data repositories. Instead it should be seen as a way for researchers to summarize data at the source for progress tracking and result sharing. Our end-goal is to enable the integration of iBIOMES Lite summaries into the iBIOMES repository. Researchers would be able to summarize their data within a computational center that does not support iRODS-based data transfers, and publish the summary into the iBIOMES repository. The raw data would not be available for download but users would be able to search for both full experiments datasets and experiment summaries via a single entry point: the repository web portal. This effort is currently supported by a common data model, a common set of parsers, and similar web interfaces.

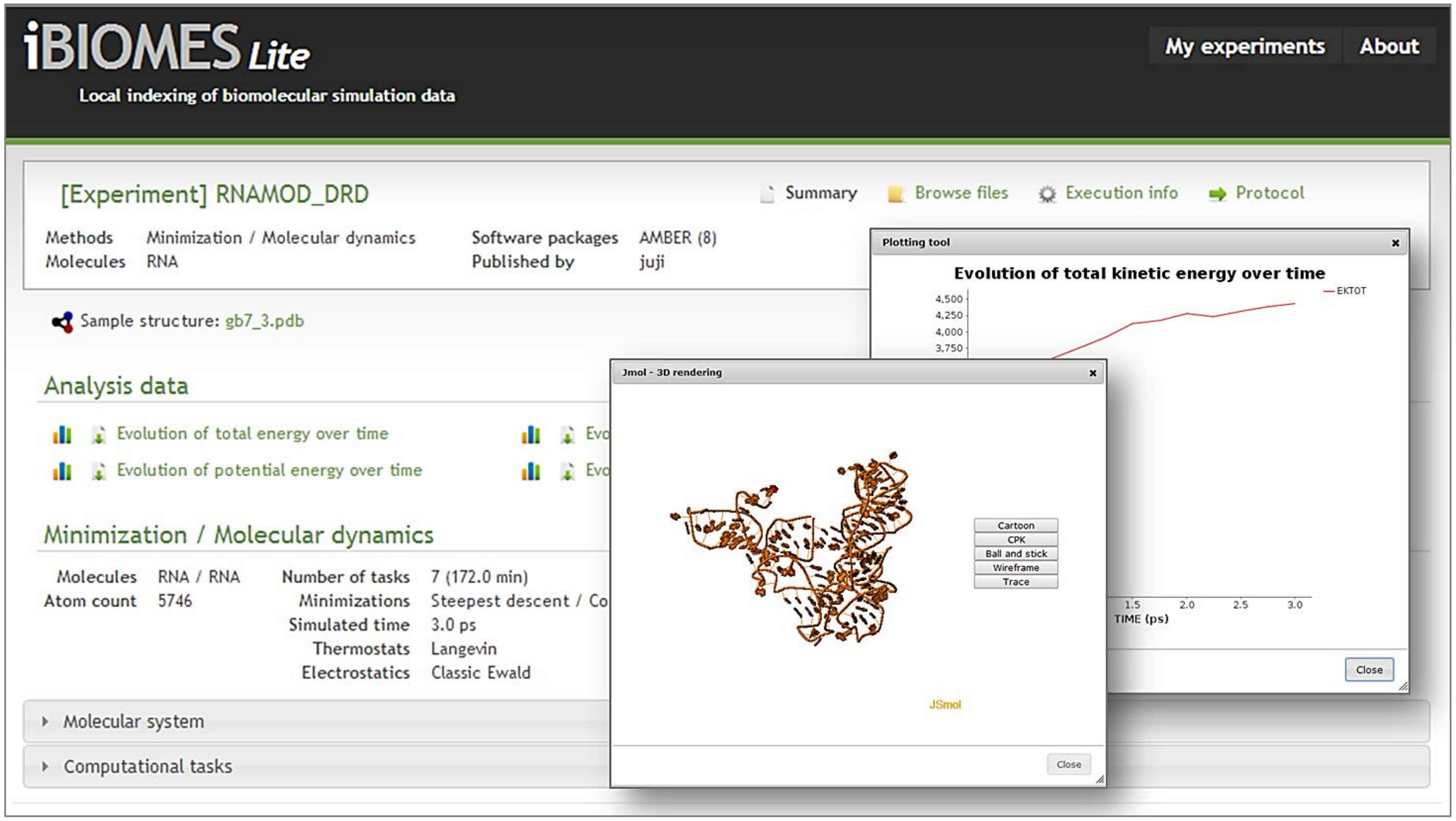
Conclusion

iBIOMES Lite provides the means for researchers to track and share biomolecular simulation datasets via automatic summarization. Summaries are supported by a software-independent data model that can describe quantum chemistry, classical and quantum MD, REMD, and QM/MM datasets. Thanks to a simple design, the tool can be easily installed on machines where users have limited privileges, whether they are hosted locally or at a national computing center. iBIOMES Lite is an open-source project and is part of the iBIOMES distribution, available at: https://github.com/jcvthibault/ibiomes.



Method	Name	Software	Molecules	Publisher	Date	Experiment path
	RNAMOD_DRD	AMBER	RNA	juji	2014-02-12 14:39	/home/juji/ibiomes/test/amber/rnamod_drd
	TUTORIAL3	AMBER	Protein	juji	2014-02-12 14:39	/home/juji/ibiomes/test/amber/tutorial3
	A-DNA	AMBER	DNA	juji	2014-02-12 14:39	/home/juji/ibiomes/test/amber/tutorial1/a-dna
	AM1-NUCLEOSIDE	AMBER	DNA	juji	2014-02-12 14:40	/home/juji/ibiomes/test/amber/sebomd/am1-nucleosi...
	MNDO-METHIONINE	AMBER	Protein	juji	2014-02-12 14:40	/home/juji/ibiomes/test/amber/sebomd/mndo-methion...
	PM3-ALANINEDIPEPTIDE	AMBER	Protein	juji	2014-02-12 14:40	/home/juji/ibiomes/test/amber/sebomd/pm3-alanined...
	ALADIP-QMMM-MD	AMBER	Protein	juji	2014-02-12 14:40	/home/juji/ibiomes/test/amber/qm_mm/aladip-qmmm-md
	REMD	AMBER	Nucleic acid	juji	2014-02-12 14:49	/home/juji/Documents/remd
	S3	GAMESS	S_3	juji	2014-02-12 14:41	/home/juji/ibiomes/test/gamess/S3
	ACETONITRILE	GAMESS	C_2H_3N	juji	2014-02-12 14:41	/home/juji/ibiomes/test/gamess/acetonitrile
	FOSFINA	GAUSSIAN	$C_{18}H_{15}P$	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gaussian/fosfina
	TAMOXIFEN	GAUSSIAN	$C_{26}H_{29}NO_2$	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gaussian/tamoxifen
	ACAC	GAUSSIAN	$C_5H_7O_2$	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gaussian/acac
	FAD	GROMACS	$C_{27}H_{10}N_9O_{15}P_2$	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gromacs/FAD
	QMMM_PROTEIN	GROMACS	Protein / $C_{18}H_4NO_6S$	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gromacs/qmmm_protein
	SPEPTIDE	GROMACS	Protein	juji	2014-02-12 14:40	/home/juji/ibiomes/test/gromacs/speptide
	NAMD-AMBER	NAMD	Protein / DNA	juji	2014-02-12 14:40	/home/juji/ibiomes/test/namd/namd-amber
	DFT_BSSE	NWChem	H_2O	juji	2014-02-12 14:40	/home/juji/ibiomes/test/nwchem/dft_bsse
	PROP_CH3F	NWChem	CH_3F	juji	2014-02-12 14:40	/home/juji/ibiomes/test/nwchem/prop_ch3f
	N2_CCSD	NWChem	N_2	juji	2014-02-12 14:41	/home/juji/ibiomes/test/nwchem/n2_ccsd

Figure 6.1, Listing of published experiments in iBIOMES Lite website.



[Experiment] RNAMOD_DRD

Methods	Minimization / Molecular dynamics	Software packages	AMBER (8)
Molecules	RNA	Published by	juji

Sample structure: gb7_3.pdb

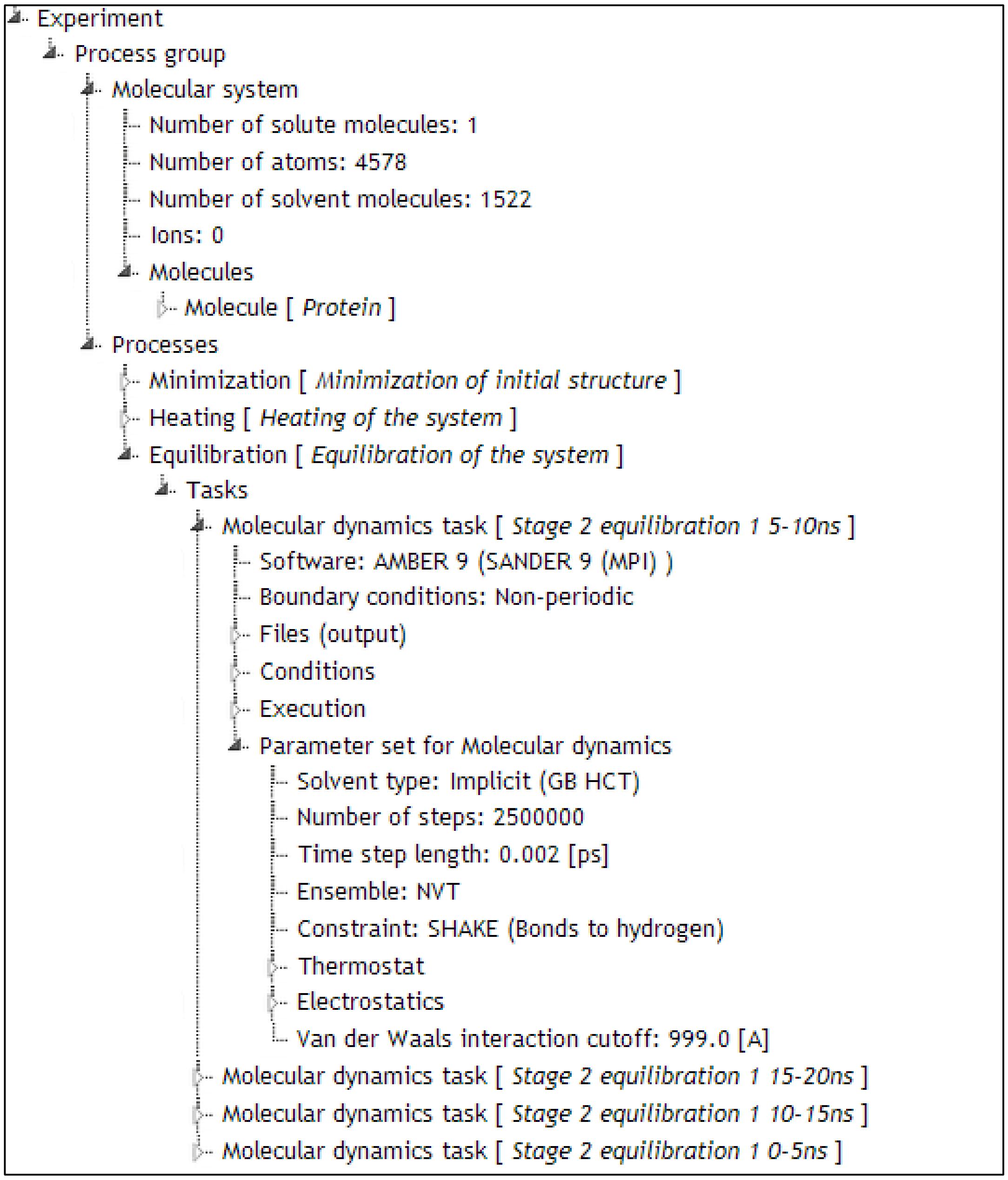
Analysis data

Evolution of total energy over time
Evo
Evolution of potential energy over time
Evo

Minimization / Molecular dynamics

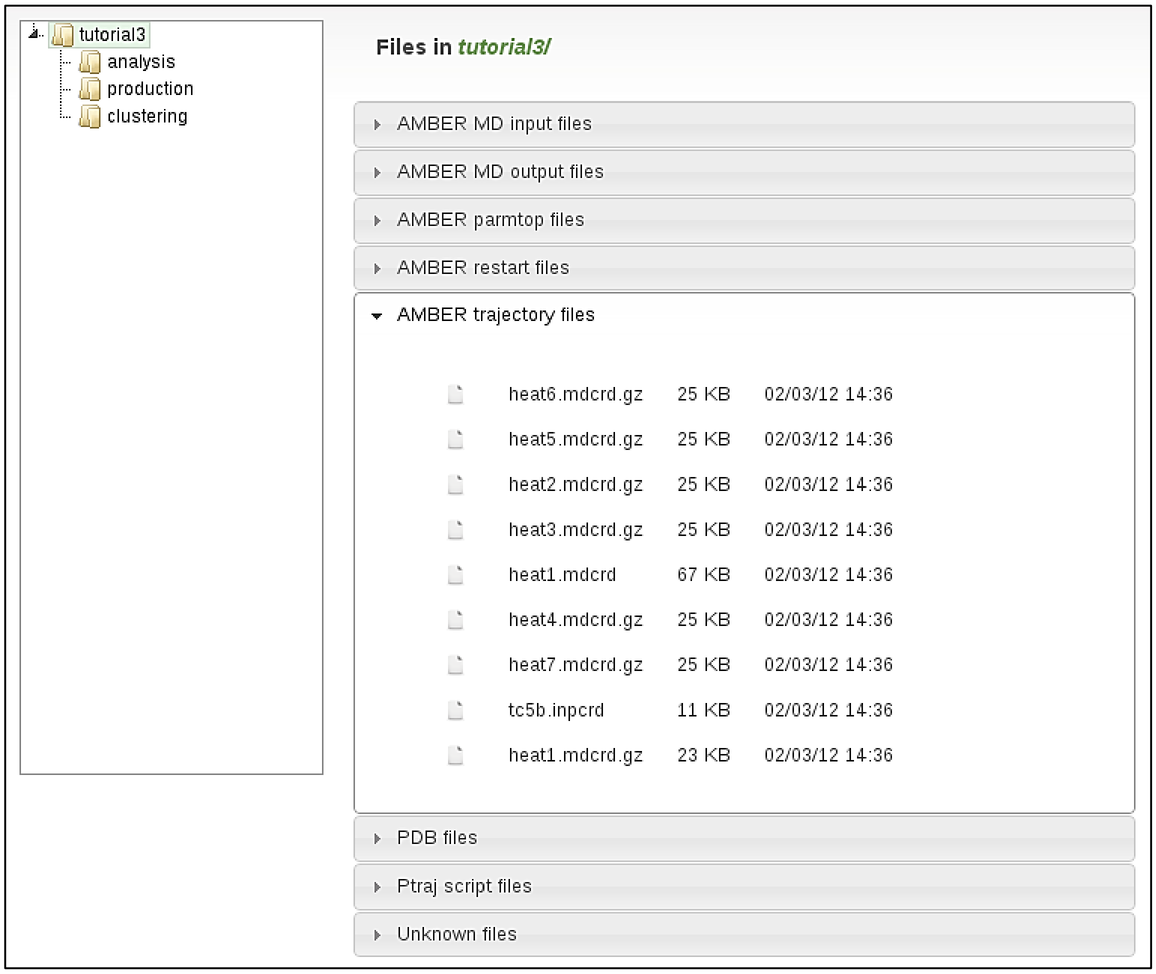
Molecules	RNA / RNA	Number of tasks	7 (172.0 min)
Atom count	5746	Minimizations	Steepest descent / Co
		Simulated time	3.0 ps
		Thermostats	Langevin
		Electrostatics	Classic Ewald

Figure 6.2, Summary of an experiment within the iBIOMES Lite website.



```
Experiment
  Process group
    Molecular system
      Number of solute molecules: 1
      Number of atoms: 4578
      Number of solvent molecules: 1522
      Ions: 0
      Molecules
        Molecule [ Protein ]
    Processes
      Minimization [ Minimization of initial structure ]
      Heating [ Heating of the system ]
      Equilibration [ Equilibration of the system ]
        Tasks
          Molecular dynamics task [ Stage 2 equilibration 1 5-10ns ]
            Software: AMBER 9 (SANDER 9 (MPI) )
            Boundary conditions: Non-periodic
            Files (output)
            Conditions
            Execution
            Parameter set for Molecular dynamics
              Solvent type: Implicit (GB HCT)
              Number of steps: 2500000
              Time step length: 0.002 [ps]
              Ensemble: NVT
              Constraint: SHAKE (Bonds to hydrogen)
              Thermostat
              Electrostatics
              Van der Waals interaction cutoff: 999.0 [A]
          Molecular dynamics task [ Stage 2 equilibration 1 15-20ns ]
          Molecular dynamics task [ Stage 2 equilibration 1 10-15ns ]
          Molecular dynamics task [ Stage 2 equilibration 1 0-5ns ]
```

Figure 6.3, Workflow details of an experiment within the iBIOMES Lite website



tutorial3
analysis
production
clustering

Files in *tutorial3/*

AMBER MD input files

AMBER MD output files

AMBER parmtop files

AMBER restart files

AMBER trajectory files

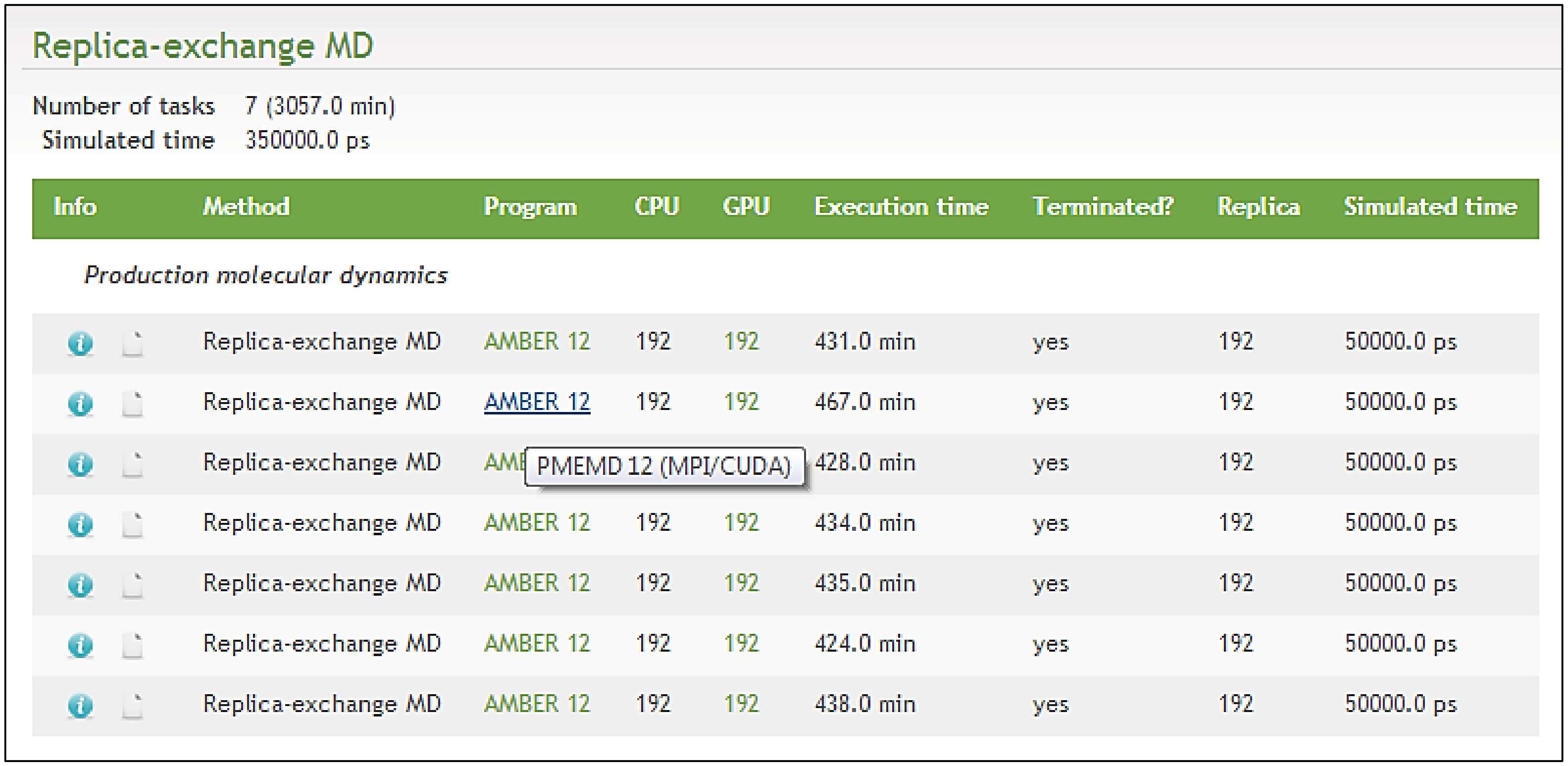
heat6.mdcrd.gz	25 KB	02/03/12 14:36
heat5.mdcrd.gz	25 KB	02/03/12 14:36
heat2.mdcrd.gz	25 KB	02/03/12 14:36
heat3.mdcrd.gz	25 KB	02/03/12 14:36
heat1.mdcrd	67 KB	02/03/12 14:36
heat4.mdcrd.gz	25 KB	02/03/12 14:36
heat7.mdcrd.gz	25 KB	02/03/12 14:36
tc5b.inpcrd	11 KB	02/03/12 14:36
heat1.mdcrd.gz	23 KB	02/03/12 14:36

PDB files

Ptraj script files

Unknown files

Figure 6.4, Experiment file listing within the iBIOMES Lite website



Replica-exchange MD

Number of tasks	7 (3057.0 min)
Simulated time	350000.0 ps

Info		Method	Program	CPU	GPU	Execution time	Terminated?	Replica	Simulated time
Production molecular dynamics									
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	431.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	467.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMI PMEMD 12 (MPI/CUDA)			428.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	434.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	435.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	424.0 min	yes	192	50000.0 ps
ℹ	🗋	Replica-exchange MD	AMBER 12	192	192	438.0 min	yes	192	50000.0 ps

Figure 6.5, Execution summary within the iBIOMES Lite website. In this example, a list of REMD runs (192 replicas each) is presented to the user with job configuration details (e.g., number of CPUs and GPUs). Extra computing environment information, such as executable details and CPU/GPU architecture, can be displayed by hovering over the associated elements.

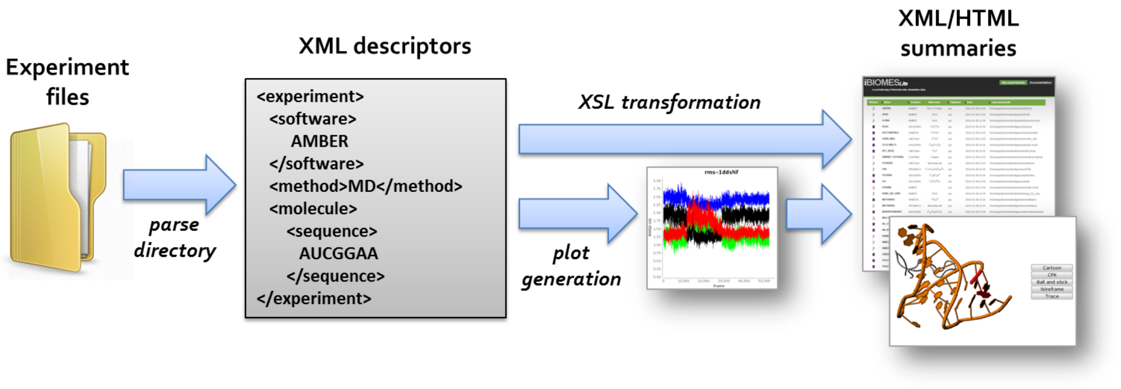


Figure 6.6, iBIOMES Lite publication process.

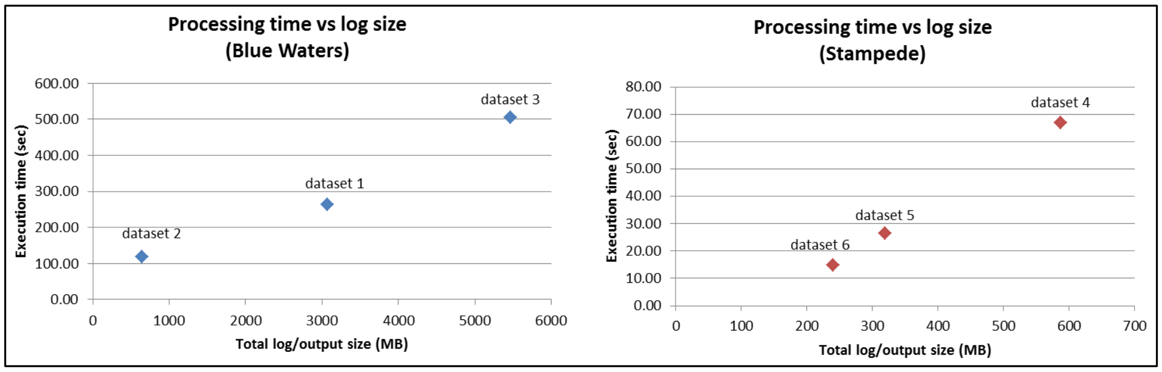


Figure 6.7, Dependence between parsing execution time and total output/log file size

Table 6.1, List of computing centers where iBIOMES Lite was successfully deployed.

Resource	Center	Description	OS	Java version
Blue Waters	NCSA	Cray XE6/XK7 system, over 25,000 nodes, including NVIDIA GK110 GPUs	UNICOS	1.7.0_07-b10
Stampede	TACC	6,400 nodes, InfiniBand Mellanox Switches/HCAs	BusyBox	1.7.0_45-b18
Gordon	SDSC	1,024 nodes, QDR InfiniBand interconnect	CentOS	1.7.0_13-b20
Ember	CHPC	262 nodes, 3144 cores, InfiniBand and Gigabit Ethernet interconnects	RHEL 6.4	1.7.0_03-b04

Table 6.2, Parsers' benchmarking on Blue Waters (NCSA) and Stampede (TACC).

Dataset	**1**	**2**	**3**	**4**	**5**	**6**
Resource	Blue Waters	Blue Waters	Blue Waters	Stampede	Stampede	Stampede
System description	RNA tetraloop	RNA tetraloop	RNA tetraloop	polymer-ligand complex	Coiled-coil dimer	Protein
Replicas/copies	192 REMD replicas	360 REMD replicas	576 REMD replicas	8 ligand configurations	5 config.	1
Number of atoms	7,622	6,071	15,599	~122,000	38,744	~22,500
Number of runs	1	1	1	147 / config.	8 / config.	12
Trajectory length*	9,600 ns	7,200 ns	17,280 ns	6,960 ns	1,000 ns	300 ns
Number of files	1,160	2,536	4,043	3,425	357	404
Total directory size	659 GB	54 GB	315 GB	816 GB	221 GB	24 GB
Log write interval	2 ps	10 ps	2 ps	10 ps	2 ps	2 ps
Average log file size	16 MB	1.8 MB	9.5 MB	0.5 MB	8 MB	20 MB
Total processed size**	3072 MB	648 MB	5472 MB	588 MB	320 MB	240 MB
Execution time						
Average (sec)	264.2	119.4	504.6	64.8	26.4	14.9
Std. dev. (sec)	58.3	2.1	43.7	1.2	0.7	0.3

*Aggregated length of all trajectories in the input folder.
**Sum of the sizes of all the MD output files in the directory.

References

1. Ng, M. H.; Johnston, S.; Wu, B.; Murdock, S. E.; Tai, K.; Fangohr, H.; Cox, S. J.; Essex, J. W.; Sansom, M. S. P.; Jeffreys, P., BioSimGrid: Grid-Enabled Biomolecular Simulation Data Storage and Analysis. *Future Gener. Comp. Sy.* **2006**, 22, 657-664.

2. Thibault, J. C.; Facelli, J. C.; Cheatham, T. E., 3rd, iBIOMES: Managing and Sharing Biomolecular Simulation Data in a Distributed Environment. *J. Chem. Inf. Model.* **2013**, 53, 726-736.

3. Rajasekar, A.; Moore, R.; Hou, C.; Lee, C. A.; Marciano, R.; de Torcy, A.; Wan, M.; Schroeder, W.; Chen, S. Y.; Gilbert, L., iRODS Primer: Integrated Rule-Oriented Data System. *Synthesis Lectures on Information Concepts, Retrieval, and Services* **2010**, 2, 1-143.

4. Vohra, S.; Hall, B. A.; Holdbrook, D. A.; Khalid, S.; Biggin, P. C., Bookshelf: A Simple Curation System for the Storage of Biomolecular Simulation Data. *Database: the Journal of Biological Databases and Curation* **2010**.

5. Goni, R.; Apostolov, R.; Lundborg, M.; Bernau, C.; Jamitzky, F.; Laure, E.; Lindhal, E.; Andrio, P.; Becerra, Y.; Orozco, M.; Lluis Gelpi, J., Standards for Data Handling. *ScalaLife White Paper* **2013**.

6. Herráez, A., Biomolecules in the Computer: Jmol to the Rescue. *Biochem. Mol. Biol. Educ.* **2006**, 34, 255-261.

7. Thibault, J. C.; Roe, D. R.; Facelli, J. C.; Cheatham III, T. E., Data Model, Dictionaries, and Desiderata for Biomolecular Simulation Data Indexing and Sharing. *J. Cheminform.* **2014**, 6, 4.

8. W3C The Extensible Stylesheet Language Family (XSL). http://www.w3.org/Style/XSL/

9. Case, D. A.; Cheatham, T. E., 3rd; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M., Jr.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J., The Amber Biomolecular Simulation Programs. *J. Comput. Chem.* **2005**, 26, 1668-1688.

10. Pronk, S.; Pall, S.; Schulz, R.; Larsson, P.; Bjelkmar, P.; Apostolov, R.; Shirts, M. R.; Smith, J. C.; Kasson, P. M.; van der Spoel, D.; Hess, B.; Lindahl, E., GROMACS 4.5: A High-Throughput and Highly Parallel Open Source Molecular Simulation Toolkit. *Bioinformatics* **2013**, 29, 845-54.

11. Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K., Scalable Molecular Dynamics with NAMD. *J. Comput. Chem.* **2005**, 26, 1781-1802.

12. Valiev, M.; Bylaska, E. J.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Van Dam, H. J. J.; Wang, D.; Nieplocha, J.; Apra, E.; Windus, T. L., NWChem: A Comprehensive and Scalable Open-Source Solution for Large Scale Molecular Simulations. *Comput. Phys. Commun.* **2010**, 181, 1477-1489.

13. Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. *Gaussian 09*, Revision C. 01; Gaussian, Inc: Wallingford, CT, 2009.

14. Saxonica *Saxon-HE (Home Edition)*, 9.5; 2014.

15. Šponer, J.; Šponer, J. E.; Mládek, A.; Banáš, P.; Jurečka, P.; Otyepka, M., How to Understand Quantum Chemical Computations on DNA and RNA Systems? A Practical Guide for Non-Specialists. *Methods* **2013**, 64, 3-11.

16. Götz, A. W.; Clark, M. A.; Walker, R. C., An Extensible Interface for QM/MM Molecular Dynamics Simulations with AMBER. *J. Comput. Chem.* **2014**, 35, 95-108.

17. Monard, G. SEBOMD (SemiEmpirical Born-Oppenheimer Molecular Dynamics): Techniques and Applications. In CECAM Workshop: Approximate Quantum-Methods: Advances, Challenges & Perspectives, University of Bremen, Germany, 2010; University of Bremen, Germany, 2010.

CHAPTER 7

DISCUSSION

Summary

Biomolecular simulation data representation

Chapter 3 introduced a common data model for biomolecular simulations. Elements described by the model cover the concepts of authorship, molecular system, computing environments, and computational method. The model introduced here is the first attempt to provide a common representation for biomolecular simulation experiments as a set of computational tasks that can use different levels of theories and different sets of parameters. The model is extensible and allows the representation of a wide variety of computational methods, including molecular dynamics, quantum chemistry and QM/MM. This model was successfully used for the design of file parsers that provide a software-independent representation of the computational experiments. Both the iBIOMES repository and iBIOMES Lite use these parsers to automatically generate common metadata and/or a logical representation of the experiments being published in these systems. The model was also used to guide the development of different prototypes, including a Grid data service that maps the logical data model to a physical database schema.

The data model was supplemented with a set of dictionaries to provide standard values and definitions for certain data elements such as computational method names. In Chapter 4 the data model and dictionaries were reorganized into a controlled vocabulary inspired by the UMLS metathesaurus. The controlled vocabulary introduces a new hierarchy between concepts and a set of semantic types to provide high-level categories for concept search and filtering. The controlled vocabulary was extended to a Simple Knowledge Organization System and a simple OWL ontology for use in a semantic web context. The ontology builds upon various OBO ontologies to enable interoperability with other popular biomedical ontologies.

Biomolecular simulation data summarization and sharing

Chapter 5 introduces the iBIOMES repository, a distributed environment to publish, index, search, and download large datasets generated by biomolecular simulations. The repository architecture builds upon the iRODS data handling system to manage files stored in distributed resources. Files and directories published in iBIOMES can be indexed using common data elements (Chapter 3) and user-specified data elements. The common data elements are defined in a separate database that includes textual descriptions and known value sets. Before a computational experiment is published into iBIOMES, the file parsers automatically extract the common data elements that summarize the experiment protocol. iBIOMES includes a web portal that can be used to build distributed queries using these data elements. Raw data can be downloaded either from the web portal or directly via the iRODS command-line interface, without prior knowledge about the physical location of the data. iBIOMES is the first open architecture for a distributed repository enabling biomolecular simulation data sharing. The system can be deployed by researchers at their own sites, independently from the computational resources that are used to generate the data. iBIOMES provides an alternative to simple centralized repositories (e.g., Bookshelf[1]) that cannot scale to a community-level approach, and to full computing environments that require users to run their simulations under a set of constraints (e.g., limited types of biomolecules, computational methods, and/or software packages). With iBIOMES, researchers do not need to adopt a new simulation workflow or give up high-performance computing resources they already have access to. Data are exposed to the repository through a simple publication process. Data exchange is enabled by indexing the raw data via common data elements while collaboration is enabled through authentication and authorization mechanisms to protect private data and open public datasets to anonymous users.

While the iBIOMES repository provides a distributed solution to biomolecular simulation data storage and indexing, other solutions are needed to manage data hosted in limited settings where the user does not have root privileges or access to IT support to deploy database components. Chapter 6 introduced iBIOMES Lite, a light-weight tool that can be deployed and used in these limited settings to summarize biomolecular simulation datasets. iBIOMES Lite is a standalone Java program that can be run in various operating systems and hardware architectures. The use of simple technology such as XSL transformation to generate HTML summaries makes it a viable solution in most environments. iBIOMES Lite was successfully deployed in various US national computing centers where big data is generated every day. Since iBIOMES Lite also uses the logical data model introduced in Chapter 3, a common representation of the data can be provided, despite the heterogeneity of the computational methods and parameters available to researchers. iBIOMES Lite is the first effort aiming at summarizing data at the source, whether it is on a personal laptop or at national computing centers with thousands of computational nodes. This tool may benefit many researchers, no matter what their IT resources are and regardless of where their data reside. Until now management systems for biomolecular simulation data have focused on creating complex infrastructures that would provide all the services necessary to run, store, analyze, and share simulations.[2, 3] Replication of such environments is not trivial because of the hardware requirements (e.g., local computational cluster, disk servers) and the IT expertise required to deploy and maintain such environments. While these integrated environments are the end-goal for simulation data management, they are currently not adapted to the distributed and heterogeneous resources researchers use. iBIOMES Lite is a simpler solution that aims to be usable by any researcher in the field, enabling data summarization, progress report, and old dataset rediscovery. As new users adopt the tool new applications for such summaries might become more obvious. The use of the raw XML summaries versus the HTML for example would provide a great solution to keep track of the provenance metadata when transferring data between institutions or when making the raw data available for download.

Limitations and future directions

The iBIOMES project

This research proposed two different architectures to satisfy researchers' needs of data indexing and sharing. On one hand iBIOMES Lite offers a simple tool that can be used by any researcher in any environment to summarize data. On the other hand the iBIOMES repository offers an infrastructure that provides a distributed solution to data storage and indexing to enable sharing and collaboration. At this point these two architectures are not interoperable. Although the same parsers are used to extract the metadata and build the logical representations of the experiments published in these environments, some work remains to be done. A long-term goal for the iBIOMES repository is to allow the publication of iBIOMES Lite-generated descriptors. The iBIOMES environment requires that the host of the data being published is already integrated in the underlying iRODS zone. If in-place registration is not possible then the raw data need to be copied to a remote iRODS-enabled server. In certain cases, neither solution will be an option. For example, if terabytes of temporary data reside at a secured computing center, it is unlikely that these data will be copied over to another resource. On the other hand the data owner might still need to keep track of these data through summaries like the ones generated by iBIOMES Lite. By allowing the publication of such summaries into the iBIOMES repository, researchers would be provided with a single end-point to track and search their datasets. Since iBIOMES Lite and the iBIOMES repository use the same parsers there is no limitation in the current architectures that would prevent such integration. For now the AVU representation would have to be chosen over the richer logical representation because of the way the iBIOMES repository indexes data. In order to store a logical representation with the same level of granularity as iBIOMES Lite a new relational database will be needed. The necessary schema has already been developed for the Grid prototype presented in Chapter 3, where logical and physical data models were mapped via Hibernate.[4] The logical representation built by the iBIOMES parsers can be persisted in this database via a Hibernate-based API module that was implemented to populate and test the Grid service prototype. Most of the remaining work will focus on the development of the web services and interfaces to query and update the relational schema. One of the advantages of the current AVU model used to tag experiment data is its simplicity: data owners can easily add, edit and remove AVU triplets, whether they are standard or user-specified attributes. One of the challenges will be to create a new mechanism that will assure consistency between the relational model and the AVU model. For example a daemon could be run to regenerate the AVU triplets on a regular basis by checking the current state of the logical model stored in the relational database. Conserving this consistency would provide two ways to query the data: either doing a keyword search (via the iRODS AVU index) or a complex query (via the relational database).

Another future direction for the iBIOMES project is the inclusion of analysis workflows as part of the data publication process. In the current versions of iBIOMES and iBIOMES Lite, analysis data can be published along with the raw data, but no mechanism is in place to assure that a minimal set of analysis tasks has been run before publication for data quality assessment. The implementation of such a mechanism will require the creation of new configuration files to define rules that will trigger alerts or actual analysis runs based on the content (i.e., file names) of the directory being published. The flags could be displayed in the current web interfaces to the data owner to provide recommendations on the analysis to run. The implementation of a process for automatic analysis of published data is more complex since it will likely require the integration of existing analysis tools and the creation of generic interfaces to wrap them into computational workflows.

Data representation

This dissertation presented two solutions to biomolecular simulation data summarization and sharing, but other approaches might be required to fulfill different requirements. For example, MDWeb[2] provides an environment to set up and run biomolecular simulations via a web portal. The resulting data are automatically stored, described, and accessed by the owner. Although this type of environment does not allow the publication of datasets generated outside the system it is well suited to newcomers to the field who might need help setting up their simulations. With the number of approaches available to researchers to run and store their data, a "one tool fits all" solution is unlikely. Therefore, one of the future challenges will be to develop data repositories and management tools that are interoperable. Creating a common data model for biomolecular simulations is a first step in this direction. In this dissertation we presented a new common data model that can represent biomolecular simulations at the experiment level, where multiple simulations and analysis tasks can be run. Although this model has already been applied to various tools, it will likely evolve as more implementations are undertaken. Nevertheless, the current model should be generic enough so that higher-level concepts such as "experiment," "task," and "parameter" will not be modified over time. On the other hand we can expect method-specific concepts to be refined and reorganized. There are several ongoing efforts in the quantum chemistry and the MD community that aim to provide a detailed description of computational task input and output.[5-7] Integration of these models into our common data model would provide a unified and rich representation of biomolecular simulations to support data exchange and interoperability. The development of a standard model for biomolecular simulation data exchange will take time and will need support from the major stakeholders, i.e., the users and the method developers. In this dissertation we presented a set of recommendations built upon community feedback and refined based on experience gained from various data exchange application implementations. These recommendations should not be taken as a new standard, but rather as a framework that will guide the development of a standard model upon which the community can agree. For example the logical model presented in Chapter 3 provides a common representation of biomolecular simulations at an abstract level, independently from any assumption about the technology. The creation of a standard will require making such assumptions to move towards syntactic interoperability. For example the definition of an XML schema will be necessary for researchers to provide stand-alone descriptors when compressing and/or moving their raw data. A standard XML schema would also enable the creation of web service interfaces on top of existing repositories that would return standard output directly reusable by external analysis or visualization tools.

Integration into the semantic web would go a step further towards interoperability. A format such as OWL, which is not domain-dependent, would allow researchers to open their data to a wider community on one hand and benefit from described and computable data sources outside their field of expertise on the other hand. In this research we presented initial work on the development of an OWL ontology that integrates popular biomedical ontologies and opens the field of biomolecular simulations to the wider field of biomedical investigations, where computational and experimental disciplines coexist. A formal evaluation of the proposed ontology is still to be done, and like the logical data model and the future data exchange formats, this will be achieved by involving groups of experts such as the Blue Obelisk[8] consortium, and developers from the MOSAIC[7] and the Scalalife[5] projects.

Conclusions

This dissertation introduced new models for the description of biomolecular simulations, a new repository architecture for the management of large datasets in a distributed environment (iBIOMES), and a light-weight tool for data summarization in limited settings (iBIOMES Lite). All these components were shown to facilitate data indexing and sharing to help researchers manage their data and collaborate within and outside the biomolecular simulation community. The data model introduced in this dissertation is the first effort to create a computable representation of the wide spectrum of computational methods used in biomolecular simulations. The two architectures based on this common representation, iBIOMES and iBIOMES Lite, not only offer solutions to the current problems faced by researchers in the field, but also an assessment of common model-driven approaches that should guide the development of future repositories.

References

1. Vohra, S.; Hall, B. A.; Holdbrook, D. A.; Khalid, S.; Biggin, P. C., Bookshelf: A Simple Curation System for the Storage of Biomolecular Simulation Data. *Database: The Journal of Biological Databases and Curation* **2010**.

2. Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Lluis Gelpi, J., MDWeb and MDMoby: An Integrated Web-Based Platform for Molecular Dynamics Simulations. *Bioinformatics* **2012**, 28, 1278-1279.

3. Simms, A. M.; Toofanny, R. D.; Kehl, C.; Benson, N. C.; Daggett, V., Dynameomics: Design of a Computational Lab Workflow and Scientific Data Repository for Protein Simulations. *Protein Eng. Des. Sel.* **2008**, 21, 369-377.

4. Hibernate. http://hibernate.org/

5. Goni, R.; Apostolov, R.; Lundborg, M.; Bernau, C.; Jamitzky, F.; Laure, E.; Lindhal, E.; Andrio, P.; Becerra, Y.; Orozco, M.; Lluis Gelpi, J., Standards for Data Handling. *ScalaLife White Paper* **2013**.

6. Phadungsukanan, W.; Kraft, M.; Townsend, J. A.; Murray-Rust, P., The Semantics of Chemical Markup Language (CML) for Computational Chemistry : CompChem. *J. Cheminform.* **2012**, 4, 15.

7. Hinsen, K., MOSAIC: A Data Model and File Formats for Molecular Simulations. *J. Chem. Inf. Model.* **2013**.

8. Guha, R.; Howard, M. T.; Hutchison, G. R.; Murray-Rust, P.; Rzepa, H.; Steinbeck, C.; Wegner, J.; Willighagen, E. L., The Blue Obelisk-Interoperability in Chemical Informatics. *J. Chem. Inf. Model.* **2006**, 46, 991-8.

APPENDIX A

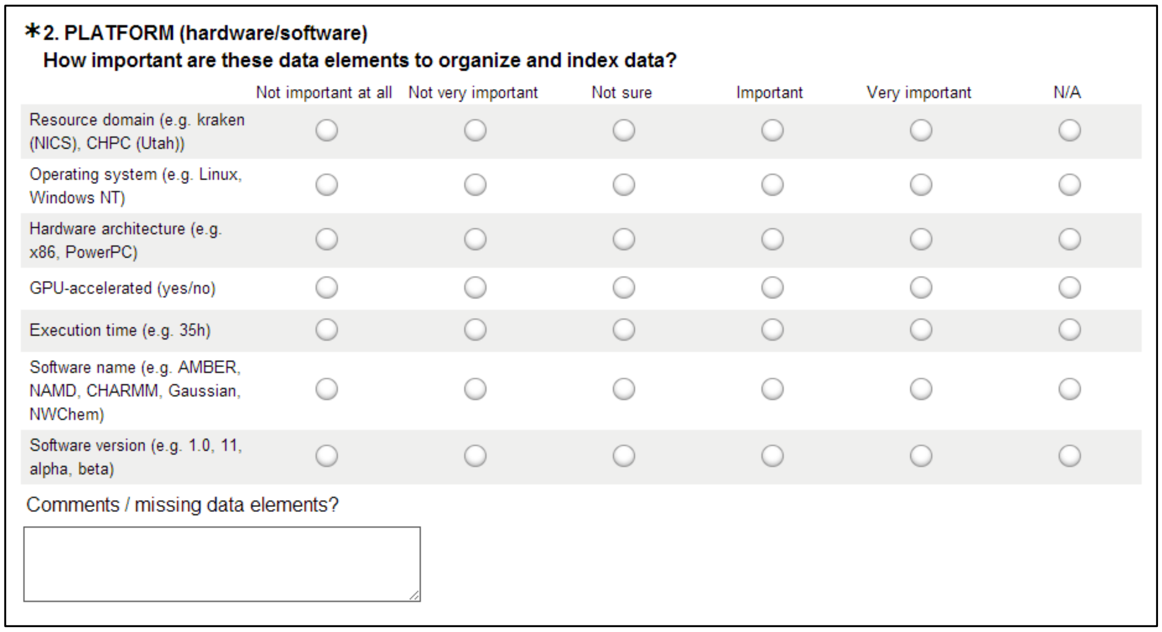
SURVEY FOR COMMON DATA ELEMENTS

Survey

Figure A.1 shows the section of the online survey that was used to assess the computational platform-related data elements. Table A.1 presents results of the survey, based on the following Likert scale: 1 = "Not important at all," 2 = "Not very important," 3 = "Not sure," 4 = "Important," 5 = "Very important," N/A = "Not applicable." N is the number of responses for a particular data element. The reported score is the average of points assigned by responders using the Likert scale. Table A.2 summarizes the comments of the respondents for each category of data elements. The last column lists only the comments that were either proposing new data elements or changes to the original ones, and that were related to the data element category. The number of respondents N is the number of people who provided at least one comment for the associated category.

Final set of common data elements

Tables A.3, A.4, A.5, A.6, A.7, and A.8 present the final list of common data elements by category. Each data element can be described through multiple attributes. Recommended attributes are marked with an "R" and attributes that can be derived from other attributes are marked with a “D”. Attributes that should be associated to a unit are marked with a “U”.



***2. PLATFORM (hardware/software)**
How important are these data elements to organize and index data?

	Not important at all	Not very important	Not sure	Important	Very important	N/A
Resource domain (e.g. kraken (NICS), CHPC (Utah))	○	○	○	○	○	○
Operating system (e.g. Linux, Windows NT)	○	○	○	○	○	○
Hardware architecture (e.g. x86, PowerPC)	○	○	○	○	○	○
GPU-accelerated (yes/no)	○	○	○	○	○	○
Execution time (e.g. 35h)	○	○	○	○	○	○
Software name (e.g. AMBER, NAMD, CHARMM, Gaussian, NWChem)	○	○	○	○	○	○
Software version (e.g. 1.0, 11, alpha, beta)	○	○	○	○	○	○

Comments / missing data elements?

Figure A.1, Online survey extract.

Table A.1, Results of the survey

Authorship data elements	Not important at all	Not very important	Not sure	Important	Very important	N/A	N	Score
Username (e.g. jthibault)	10	12	6	6	3	0	37	2.46
Full name (e.g. Julien Thibault)	3	2	3	12	19	0	39	4.08
Institution Name (e.g. University of Utah)	5	3	3	16	11	0	38	3.66
E-mail (e.g. julien.thibault@utah.edu)	3	3	4	16	12	1	39	3.72
Publication that is related to the current experiment (e.g. URL, DOI)	0	2	4	13	19	1	39	4.18
Publication that is based on the results of this run (e.g. URL, DOI)	0	2	0	15	21	1	39	4.33
Platform data elements	**Not important at all**	**Not very important**	**Not sure**	**Important**	**Very important**	**N/A**	**Responses**	**Score**
Resource domain (e.g. kraken (NICS), CHPC (Utah))	5	14	8	9	3	0	39	2.77
Operating system (e.g. Linux, Windows NT)	4	12	4	13	6	0	39	3.13
Hardware architecture (e.g. x86, PowerPC)	2	11	2	18	6	0	39	3.38
GPU-accelerated (yes/no)	3	7	5	15	9	0	39	3.51
Execution time (e.g. 35h)	3	10	5	12	9	0	39	3.36
Software name (e.g. AMBER, NAMD, CHARMM, Gaussian, NWChem)	0	0	0	4	34	1	39	4.77
Software version (e.g. 1.0, 11, alpha, beta)	0	2	0	6	30	1	39	4.56
Molecular system data elements	**Not important at all**	**Not very important**	**Not sure**	**Important**	**Very important**	**N/A**	**Responses**	**Score**
Composition of the solvent (e.g. Water, Na+)	0	1	1	7	29	1	39	4.56
Number of water molecules in the system	1	2	0	12	22	2	39	4.18
Number of atoms in the system	0	1	1	9	27	1	39	4.51
Number of ions in the system	0	3	1	9	24	2	39	4.23
Molecule type (e.g. Protein, RNA, DNA, chemical compound, nano-particle)	1	1	1	5	30	1	39	4.51
Molecule name (e.g. Alanine, Sucrose, Tamoxifen)	1	2	2	12	21	1	39	4.21
Sequence (Amino-acid or nucleotide sequence)	1	0	1	15	21	1	39	4.33
Reference structure (e.g. PDB:1BIV, PubChem:2733526)	0	0	1	13	24	1	39	4.49
Molecular formula (e.g. C26H29NO)	1	7	4	14	12	1	39	3.67
Molecular weight (e.g. 371.51456 g/mol)	1	16	7	9	5	1	39	2.95

Table A.1, Continued

Computational method data elements	Not important at all	Not very important	Not sure	Important	Very important	N/A	Responses	Score
General method name (e.g. Molecular dynamics, QM, Coarse-grain Dynamics, QM/MM)	0	0	0	8	30	1	39	4.67
Method reference citation (e.g. DOI, URL)	2	8	8	9	11	1	39	3.41
Whether the method simulate the dynamics of the system (Yes / No)	1	2	8	12	15	1	39	3.90
Type of boundary conditions (Periodic, non-periodic)	0	1	3	6	28	1	39	4.49
Whether the run has converged (yes/no)	3	2	7	8	16	3	39	3.59
Convergence criteria (e.g. 10^-3)	3	4	5	8	17	2	39	3.67
Representation of the solvent (implicit, explicit, in vacum)	1	0	1	6	30	1	39	4.56
Molecular Dynamics data elements	**Not important at all**	**Not very important**	**Not sure**	**Important**	**Very important**	**N/A**	**Responses**	**Score**
Force field (e.g. AMBER FF 99, GROMOS 43A1 , ReaxFF)	0	0	0	5	33	1	39	4.74
Force field type (e.g. classical, polarizable, reactive)	0	1	1	14	22	1	39	4.38
Unit shape (e.g. cuboid, octahedron, cap, shell)	0	8	4	12	14	1	39	3.74
Ensemble type (e.g. NVE, NVT, NPT, Generalized)	0	3	2	10	23	1	39	4.28
Barostat (e.g. Andersen, Berendsen, Parrinello-Rahman)	1	7	1	14	14	1	38	3.79
Barostat time constant (e.g. 1000 fs)	1	8	3	16	10	1	39	3.59
Thermostat (e.g. Berendsen, Nose, Nose-Poincare)	1	5	3	14	14	1	38	3.84
Thermostat time constant (e.g. 100 fs)	1	7	3	17	10	1	39	3.64
Molecular mechanics integrator (e.g. Euler, Runge-Kutta, Verlet, Leapfrog)	1	11	3	15	8	1	39	3.38
Constraint algorithm (e.g. LINCS, RATTLE, SHAKE, SETTLE)	1	7	3	16	11	1	39	3.67
Electrostatics modeling (e.g. Cutoff, Classic ewald, PME, reaction field)	0	3	3	7	24	1	38	4.29
Time step length (e.g. 1 picosecond)	1	3	2	13	19	1	39	4.10
Total simulated time (e.g. 450 picoseconds)	0	0	0	9	29	1	39	4.64

Table A.1, Continued

Quantum Mechanics data elements	Not important at all	Not very important	Not sure	Important	Very important	N/A	Responses	Score
Category of QM method (e.g. Hartree-Fock, Moeller-Plesset, DFT, Configuration Interaction)	0	0	2	3	30	4	39	4.31
Level of theory (e.g. SCF, MP2, MP4, CCSD(T))	0	0	2	1	32	4	39	4.36
Basis set (e.g. STO-3G, 6-31++G*, cc-pCDVZ)	0	0	3	2	30	4	39	4.28
Basis set family (e.g. minimal, Pople, correlation consistent)	1	7	7	8	12	4	39	3.28

Table A.2, Summary of survey comments for each data element category

Data element category	N	Proposed data elements and changes
Authorship	4	- Missing: grant information - Missing: timestamp / upload date
Platform (hardware/software)	4	- Missing: software compiled in single or double precision - Change: GPU-accelerated is part of hardware architecture - Missing: memory requirement, problems encountered during run
Molecular system	5	- Change: number of water molecules should be number of solvent molecules - Missing: rigid parameters (e.g. some coordinate) - Missing: water model is important
Molecule	5	- Missing: apparent pH - Missing: information about the ligand (geometry and parameters) - Missing: important functional groups
Method (all)	7	- Missing: broad classification of methods (empirical, semi-empirical, DFT, ab initio or combo of these) as well as static vs. dynamic. - Change: convergence is both case dependent (energy vs. entropy vs. heat capacity...), and is also quite subjective. - Change: convergence criteria would be difficult to track as the user will decide how to judge this - Change: convergence is a moving target at best. Maybe there should be an overall convergence criteria metric, and if this minimum is met, it could be filed under "converged."
MD methods	6	- Missing: advanced sampling details, output details (e.g. steps per write), simulation scheme (whether this was a production run with such and such minimization and equilibration) - Missing: restraints - Missing: for PME, order of interpolation. For LINCS, order of expansion of the series. - Missing: parallelization scheme
QM methods	4	- Missing: general property classifications (e.g. electron properties, pseudopotentials, frozen core) - Missing: set of output properties available, and if QM method uses density functional theory related choices of exchange correlation and cut-offs

Table A.3, Data elements related to authorship

Authorship (scope: experiment)	Attribute	U	R	D
Author	Full name (e.g. John Doe)		R	
	Institution name (e.g. university, company)			
	E-mail (e.g. john.doe@my.university.edu)			
Citation	Identifier (e.g. DOI, PubMed ID)		R	
	URL			
Publication based on the experiment results	Identifier (e.g. DOI, PubMed ID)		R	
	URL			
Grant	Identifier		R	
	Source			
	Title			

Table A.4, Data elements related to the computational platform (hardware/software)

Platform (scope: task)	Attribute	U	R	D
Computational environment	Resource domain (e.g. Kraken (NICS), Gordon (SDSC))			
	Machine/supercomputer architecture (e.g. Cray XK7, IBM Blue Gene/Q)			
	Operating system (e.g. Linux, Windows NT)			
	CPU architecture (e.g. x86, PowerPC)			
	GPU architecture (e.g. Nvidia GTX 780)			
Execution	Execution time (e.g. 35h)	U		
	Normal termination		R	
	Number of CPUs used			
	Number of GPUs used			
Software	Name (e.g. AMBER, NAMD, CHARMM, Gaussian, NWChem)		R	
	Version (e.g. 1.0, 11, alpha, beta)		R	

Table A.5, Data elements related to the molecular system definition

Molecular system	Attribute	U	R	D
System	Composition of the solvent (e.g. Water, Na+)		R	
	Number of solute molecules		R	
	Number of solvent molecules		R	
	Number of atoms in the system		R	
	Number of ions in the system		R	
	Apparent pH			
Molecule	Type (e.g. Protein, RNA, DNA, chemical compound, nano-particle)		R	
	Name (e.g. Alanine, Sucrose, Tamoxifen)		R	
	Residue sequence (Amino-acid or nucleotide sequence)		R	
	Reference structure (e.g. PDB:1BIV, PubChem:2733526)		R	
	Molecular formula (e.g. C26H29NO)			
	Molecular weight (e.g. 371.51456 g/mol)	U		
	Whether it is part of the solvent or the solute		R	
	Main functional groups			

Table A.6, Data elements common to any type of computational method

Method (scope: task)	Attribute	U	R	D
Method	General method name (e.g. MD, QM, Coarse-grain Dynamics, QM/MM)		R	
	Method reference citation (e.g. DOI, URL)			
	Whether the method simulates the dynamics of the system (Yes / No)			
Boundary conditions	Type (Periodic, non-periodic)		R	
Solvent model	Representation of the solvent (implicit, explicit, in vacuum)		R	
	Implicit solvent model name (e.g. GB HCT)			

Table A.7, Data elements specific to molecular dynamics

MD (scope: task)	Attribute	U	R	D
Electrostatics model	Name (e.g. Cutoff, Classic ewald, PME, reaction field)		R	
Unit shape	Type (e.g. cuboid, octahedron, cap, shell)			
Ensemble	Type (e.g. NVE, NVT, NPT, Generalized)		R	
Molecular mechanics integrator	Name (e.g. Euler, Runge-Kutta, Verlet, Leapfrog)			
Constraint	Algorithm (e.g. LINCS, RATTLE, SHAKE, SETTLE)			
	Target			
Restraint	Type (e.g. bond, angle)			
	Target			
Force field	Name (e.g. AMBER FF 99, GROMOS 43A1 , ReaxFF)		R	
	Type (e.g. classical, polarizable, reactive)		R	D
Barostat	Name (e.g. Andersen, Berendsen, Parrinello-Rahman)			
	Time constant (e.g. 1000 fs)	U		
Thermostat	Name (e.g. Berendsen, Nose, Nose-Poincare)			
	Time constant (e.g. 100 fs)	U		
Time	Time step length (e.g. 1 picosecond)	U	R	
	Number of time steps		R	
	Total simulated time (e.g. 450 picoseconds)	U	R	D
Context of the run	Type (minimization, equilibration, or production)			
Enhanced sampling method	Name (e.g. umbrella sampling, replica-exchange)			

Table A.8, Data elements specific to quantum chemistry

QM (scope: task)	Attribute	U	R	D
QM method	Specific name (e.g. SCF, MP2, MP4, CCSD(T), B3LYP)		R	
	Family (e.g. Hartree-Fock, Moeller-Plesset, DFT, Configuration Interaction)		R	D
Basis set	Name (e.g. STO-3G, 6-31++G*, cc-pCDVZ)		R	
	Family (e.g. minimal, split-valence, plane-wave)			D
Spin multiplicity	Value			
Total charge	Value			
Froze core	Uses frozen core (yes/no)			
Pseudo-potential	Implementation name (e.g. Martins-Trouiller)			
	Plane-wave cutoff	U		
Convergence	Whether the run has converged (yes/no)			
	Convergence criteria (e.g. 10^-3)	U		
Exchange-correlation functional	Name (e.g. B3LYP)			

APPENDIX B

COMMON REPRESENTATION FOR ANALYSIS DATA: EXAMPLES

Two examples of how the proposed data elements might be applied to common analysis data will be given. Note that currently the programs used in these examples do not necessarily report all of the metadata for these attributes; rather this is a recommendation of what metadata these programs could include in their output.

The first example is the calculation of a distance between two atoms in a protein over the course of a molecular dynamics simulation totaling 101 ps in length, with the trajectory recorded at 1 frame per ps. The generated data set metadata can be as follows:

```
Analysis Name: Distance
Description: Distance in Cartesian space.
File: end-to-end.dat
Timestamp: Sat Nov 30 09:49:37 MST 2013
Filter on space: (Residue 2 atom CA), (Residue 12 atom CA)
Number Data Set Dimensions: 1
      Dimension[1] size: 101
Number of variables: 2
      Variable[1] units: picosecond
      Variable[1] label: Time
      Variable[1] type: float
      Variable[1] uses dimension: 1
      Variable[2] units: Angstrom
      Variable[2] label: End to end distance
      Variable[2] type: float
      Variable[2] uses dimension: 1
Program: VMD
      Version: V1.9.1
      Command: distance “resid 2 and name CA” “resid 12 and name
CA” 1 end-to-end.dat distr.dat
```

Note that there are actually two arrays sharing the same dimension, one (‘End to end distance’) containing the distance data and another (‘Time’) that holds the corresponding time steps of the data.

The next example is the calculation of a mass-weighted coordinate covariance matrix for C-alpha atoms (12 atoms total) over 10 frames. Again there are two variables, but in this case the ‘Time’ variable would record which frames were used in generating the matrix, while the ‘matrix1’ variable is the 12x12 matrix itself.

```
Name: Mass-weighted Covariance Matrix
File: mwcovar.dat
Timestamp: Sat Nov 30 09:58:22 MST 2013
Filter on space: (All CA atoms)
Number Data Set Dimensions: 3
      Dimension[1] size: 12
      Dimension[2] size: 12
      Dimension[3] size: 10
Number of variables: 2
      Variable[1] units: picosecond
      Variable[1] label: Time
      Variable[1] type: float
      Variable[1] uses dimension: 3
      Variable[2] units: Angstrom*amu^0.5
      Variable[2] label: matrix1
      Variable[2] type: float
      Variable[2] uses dimensions: 1, 2
Program: Cpptraj
      Version: V13.12
      Command: matrix mwcovar out mwcovar.dat name matrix1
```

APPENDIX C

DICTIONARY EXAMPLES

Table C.1 lists a few force field parameter sets available for popular MD software packages. Each entry in the table is described through an ID (ID), a name (TERM), a description (DESCRIPTION), a possible list of citations (CITATION), a force field type ID (TYPE_ID), and whether the force field is coarse grain or not (IS_COARSE_GRAIN).

Table C.2 lists "specific" methods which can be referenced within an input file for a computational task. Each entry in the table is described through an ID (ID), a name (TERM), a description (DESCRIPTION), and a possible list of citations (CITATION).

Table C.1, Extract from the force field dictionary.

ID	TERM	DESCRIPTION	CITATION
10	AMBER FF10	AMBER FF10 force field	AMBER Tools 10 manual. Available at: http://ambermd.org/doc10/AmberTools.pdf
11	AMBER FF12SB	AMBER FF12SB force field	AMBER Tools 12 manual. Available at: http://ambermd.org/doc12/AmberTools12.pdf
12	AMBER GAFF	General Amber Force Field (GAFF) for small molecules	Wang, J.; Wolf, R.M.; Caldwell, J.W.; Kollamn, P.A.; Case, D.A. Development and testing of a general Amber force field. J. Comput. Chem., 2004, 25, 1157-1174
50	CHARMM 19	CHARMM 19 force field	Reiher, III WH (1985). 'Theoretical studies of hydrogen bonding'. PhD Thesis at Harvard University.
51	CHARMM 22	CHARMM 22 force field	MacKerell, Jr. AD, et al. (1998). 'All-atom empirical potential for molecular modeling and dynamics studies of proteins'. J Phys Chem B 102 (18): 3586-3616.
52	CHARMM 27	CHARMM 27 force field	MacKerell, Jr. AD, Banavali N, Foloppe N (2001). 'Development and current status of the CHARMM force field for nucleic acids'. Biopolymers 56 (4): 257-265.

Table C.2, Extract from the dictionary of computational methods.

ID	TERM	DESCRIPTION	CITATION
1	HF	Hartree-Fock	-
2	UHF	Unrestricted Hartree-Fock	-
3	ROHF	Restricted open-shell Hartree-Fock	-
4	SCF	Self-consistent field	-
5	MP2	Moeller-Plesset perturbation theory (second-order)	-
6	MP3	Moeller-Plesset perturbation theory (third-order)	-
7	MP4	Moeller-Plesset perturbation theory (fourth-order)	-
8	MP5	Moeller-Plesset perturbation theory (fifth-order)	-
9	CISD	Configuration interaction singles and doubles	-
10	CISDT	Configuration interaction singles, doubles, and triples	-
11	CISDTQ	Configuration interaction singles, doubles, triples, and quadruples	-
12	CCD	Coupled-cluster doubles	-
13	CCSD	Coupled-cluster singles and doubles	-

APPENDIX D

LUCENE-BASED DICTIONARY USAGE

AND LOOKUP EXAMPLE

Usage

```
lucene-lookup.sh [options]

Options:

lookup -i <index-path> -t <term> [-f <lookup-field>] [-n <max-hits>]
list   -i <index-path>

lookup: look up a term <term> in the Lucene index at <index-path> in a
particular field <lookup-field>.
list: lists all the entries in the Lucene index at <index-path>
```

Example

Input command

```
lucene-lookup.sh lookup -i /tmp/dictionary_all -t "AMBER FF*" -n 2
```

Console output

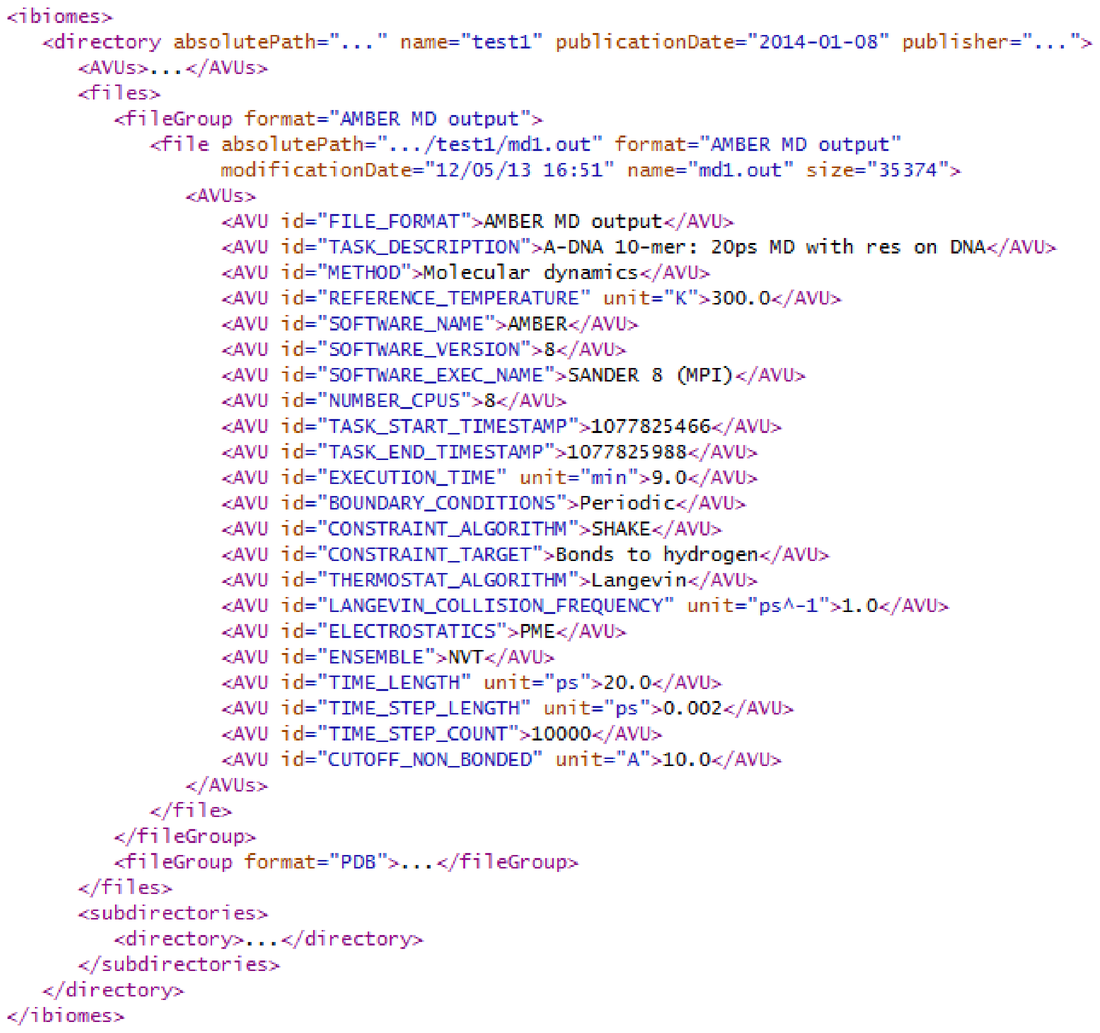
```
Lookup field: TERM
        Term: AMBER FF*
    Max hits: 2
  Dictionary: /tmp/dictionary_all
Number of entries: 939

2 matches:
--------------------------------
  [UID] 885
  [ID] 1
  [TERM] AMBER FF94
  [DESCRIPTION] AMBER FF94 force field
  [CITATION] Cornell et al. (1995), JACS 117, 5179-5197
  [TYPE_ID] 1
  [IS_COARSE_GRAIN] No
  [ATTRIBUTE_TYPE] force_field
--------------------------------
  [UID] 886
  [ID] 2
  [TERM] AMBER FF96
  [DESCRIPTION] AMBER FF96 force field
  [CITATION] Kollman (1996), Acc. Chem. Res. 29, 461-469
  [TYPE_ID] 1
  [IS_COARSE_GRAIN] No
  [ATTRIBUTE_TYPE] force_field
--------------------------------
```

APPENDIX E

XML REPRESENTATIONS FOR SIMULATION DATA INDEXING

Figure E.1 presents an example of the XML representation that describes the file tree associated to a given computational experiment (physical view), in this case a short MD simulation of a DNA 10-mer helix. Each file is associated to a list of AVUs (Attribute-Value-Units) for indexing. Figure E.2 present an example of the XML representation of the experimental protocol (logical view) associated to the same computational experiment.



```
<ibiomes>
   <directory absolutePath="..." name="test1" publicationDate="2014-01-08" publisher="...">
      <AVUs>...</AVUs>
      <files>
         <fileGroup format="AMBER MD output">
            <file absolutePath=".../test1/md1.out" format="AMBER MD output"
                  modificationDate="12/05/13 16:51" name="md1.out" size="35374">
               <AVUs>
                  <AVU id="FILE_FORMAT">AMBER MD output</AVU>
                  <AVU id="TASK_DESCRIPTION">A-DNA 10-mer: 20ps MD with res on DNA</AVU>
                  <AVU id="METHOD">Molecular dynamics</AVU>
                  <AVU id="REFERENCE_TEMPERATURE" unit="K">300.0</AVU>
                  <AVU id="SOFTWARE_NAME">AMBER</AVU>
                  <AVU id="SOFTWARE_VERSION">8</AVU>
                  <AVU id="SOFTWARE_EXEC_NAME">SANDER 8 (MPI)</AVU>
                  <AVU id="NUMBER_CPUS">8</AVU>
                  <AVU id="TASK_START_TIMESTAMP">1077825466</AVU>
                  <AVU id="TASK_END_TIMESTAMP">1077825988</AVU>
                  <AVU id="EXECUTION_TIME" unit="min">9.0</AVU>
                  <AVU id="BOUNDARY_CONDITIONS">Periodic</AVU>
                  <AVU id="CONSTRAINT_ALGORITHM">SHAKE</AVU>
                  <AVU id="CONSTRAINT_TARGET">Bonds to hydrogen</AVU>
                  <AVU id="THERMOSTAT_ALGORITHM">Langevin</AVU>
                  <AVU id="LANGEVIN_COLLISION_FREQUENCY" unit="ps^-1">1.0</AVU>
                  <AVU id="ELECTROSTATICS">PME</AVU>
                  <AVU id="ENSEMBLE">NVT</AVU>
                  <AVU id="TIME_LENGTH" unit="ps">20.0</AVU>
                  <AVU id="TIME_STEP_LENGTH" unit="ps">0.002</AVU>
                  <AVU id="TIME_STEP_COUNT">10000</AVU>
                  <AVU id="CUTOFF_NON_BONDED" unit="A">10.0</AVU>
               </AVUs>
            </file>
         </fileGroup>
         <fileGroup format="PDB">...</fileGroup>
      </files>
      <subdirectories>
         <directory>...</directory>
      </subdirectories>
   </directory>
</ibiomes>
```

Figure E.1. XML representation of the file tree associated to a computational experiment

```xml
<experiment name="test1" publicationDate="2014-01-08" publisher="..." rootDirectoryPath=".../test1">
   <processGroups>
      <processGroup>
         <molecularSystem>
            <soluteMolecules>
               <molecule atomCount="319" residueCount="10" type="DNA">...</molecule>
            </soluteMolecules>
         </molecularSystem>
         <processes>
            <process name="Minimization">
               <description>Minimization of initial structure</description>
               <tasks>...</tasks>
            </process>
            <process name="Production MD">
               <description>Production molecular dynamics</description>
               <tasks>
                  <task type="Molecular dynamics">
                     <boundaryConditions>Periodic</boundaryConditions>
                     <description>A-DNA 10-mer: 20ps MD with res on DNA</description>
                     <outputFiles><file>.../test1/md1.out</file></outputFiles>
                     <simulatedConditionSet><referenceTemperature unit="K" value="300.0"/></simulatedConditionSet>
                     <software executableName="SANDER 8 (MPI)" name="AMBER" version="8"/>
                     <taskExecution normalTermination="true">...</taskExecution>
                     <parameterSet type="Molecular dynamics">
                        <constraints><constraint algorithm="SHAKE" target="Bonds to hydrogen"/></constraints>
                        <cutoffForNonbondedInteractions unit="A" value="10.0"/>
                        <electrostaticsModel><name>PME</name></electrostaticsModel>
                        <ensemble>NVT</ensemble>
                        <numberOfSteps>10000</numberOfSteps>
                        <simulatedTime unit="ps" value="20.0"/>
                        <thermostat name="Langevin"><collisionFrequency unit="ps^-1" value="1.0"/></thermostat>
                        <timeStepLength unit="ps" value="0.002"/>
                     </parameterSet>
                  </task>
               </tasks>
            </process>
         </processes>
      </processGroup>
   </processGroups>
   <analysis>
      <spreadsheets>
         <spreadsheet description="Kinetic energy for the center of mass in translation over time" path="..."/>
   </analysis>
</experiment>
```

Figure E.2, XML representation of the computational experiment protocol.