Chore: Molecular Dynamics Case Study

Parker Diamond, Elliot Greenlee, Jason Liang
Department of Electrical Engineering and Computer Science
University of Tennessee
Knoxville, TN 37996
jdiamon3@vols.utk.edu
egreenle@vols.utk.edu
jliang7@vols.utk.edu

Abstract

This report details the design and implementation of a bounty board system to complete computationally intensive research jobs using distributed resources. As computational requirements increase for researchers, a low cost solution is needed that rewards users. Our prototype of the system demonstrates that molecular dynamics simulations fit the distributed and security requirements of the work system. In order to fit both bioinformatics and data mining perspectives, we will approach the system using the Pathogen Box dataset, and attempt to simulate aggregations in water of the various drug molecules.

1 Introduction

Right now data generation is surging exponentially, while growth in computational power is slowing. In order to manage these quantities of information and provide use, brute force simulations, experiments, and algorithms are applied by scientists and engineers. Unfortunately, for many professionals more computation is needed than provided. For companies, this means development in cloud based solutions like Amazon Web Services, which provide efficient and scalable tools for any solution. Alternatively, organizations can build dedicated clusters like Anton[8], which can simulate 17,000 nanoseconds of simulation time per day for a 24,000 atom system. However, researchers need minimally expensive or free solutions to match their funding.

One solution to this is outsourcing computational requirements to the public. Currently, other systems exist that employ widely distributed minimal computer resources for computation by exploiting idle time from a wide user base. In the domain of molecular dynamics for protein folding, Folding@home and CureCoin focus on using the unused computer power of individual users. In the case of Folding@home by Stanford, altruistic individuals volunteer idle time on their computers towards molecular dynamics. The project, launched publicly by Pande Lab in 2000, has explored diseases from malaria to Alzheimer's disease in a directed effort spanning two decades. In 2016 the group reached estimated 100 petaflops of computational power [1]. However, altruism is not a strong enough motivator for most users, and does not attract larger computational resources to support growth. Curecoin, on the other hand tries to encourage folding on the Folding@home system using a new cryptocurrency to provide value [4]. Users mine the currency by completing simulations in an idle browser window. Unfortunately, cryptocurrencies have values which fluctuate rapidly, have strong natural limits placed on use beyond peer to peer transactions, and are challenging to understand for the broader public.

We propose a paid system of computational bounties. Researchers may submit jobs to a large queue for processing, either as a free job or as an escalated paid job. The more a job is worth, the more likely a user is to claim and complete it. A taxed percentage of funds from each researcher is also allocated to all other jobs in the system, such that initially free jobs slowly accrue value over time, increasing the likelihood that they will be completed.

This system requires multiple security and integrity conditions. First, a central authority must evaluate and approve each job submitted by the researcher to establish researcher-worker trust and calculate the job value against a standard work hours unit. Users are limited based on a maximum time it would take them to complete a certain job submitted. For each type of job submitted the system must ensure proof of work, result integrity, and worker security. In this paper, we evaluate molecular dynamics simulations as a job for the system.

Subtropical regions in poverty and in close contact with many infectious vectors are at risk to sundry neglected tropical diseases (NTDs). Each year, more than one billion people are affected and billions of dollars are lost in developing economies. The Pathogen Box provides many drug-like compounds relevant to these NTDs for study in order to catalyze research. This case study takes the perspective of a researcher galvanized to perform molecular dynamics simulations and modeling of the molecules, specifically their aggregation behavior in water. This research is useful for all new drug types, since drugs should distribute in the blood rather than coagulate.

2 System

2.1 Bounty System

In this section, we abstract molecular dynamics simulations as jobs that are submitted by researchers, and describe the system as a user story from start to finish of a single job.

The Chore system, visualized in Figure 1 is made up of two components: a server run by a central authority in charge of the system, and an easy to use client run without expert knowledge. The server system is set up in Flask using an SQLAlchemy database to store information about both Users and Jobs.

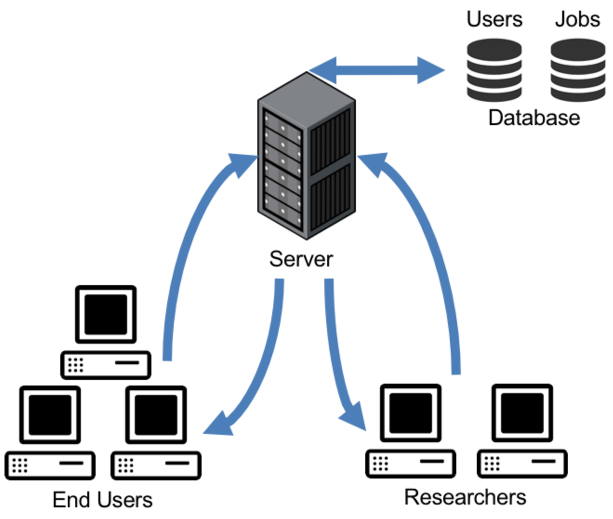


Figure 1: The full Chore system.

Users include both researchers and end users. When a user registers, they send both the details of their payment account and their computer system, both of which are verified by the central authority and the client. At this point, the system creates an ID number, fills in their account value, and adds the user to the database. This information is used differently depending on if the user is a researcher or and end user. See Table 1 for a description of user data.

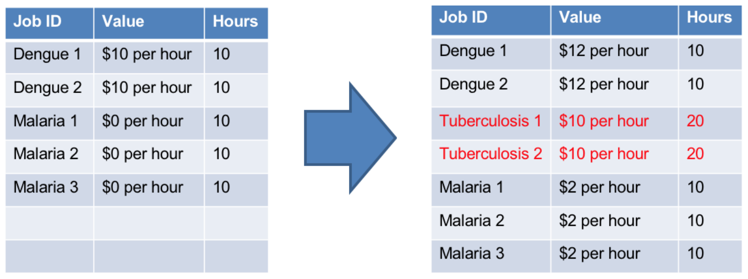
Table 1: User data fields.

Field	Description
id	to identify the user
money_id	bank/paypal account
money	money in the account
completed_jobs	number of jobs completed by a user
submitted_jobs	number of jobs submitted by a researcher
computer_rating	user computer performance relative to a standard

First, a researcher submits a job by sending the job files and code, the time it takes to run the job, and how much money to spend on the job. At this point, the system creates an ID Number, stores the jobs files appropriately, and calculates the standard job time by multiplying the researcher computer rating by the time it takes to run the job, before adding the job to the database. The client keeps track of the researchers specific jobs. See Table 2 for a description of job data. The system also taxes the job at a constant rate, and distributes the money equally to all other jobs. See Figure 2 for an example of two system states, one previous to and one after a researcher submits two jobs for a total of $500, which is being taxed at 20% and distributed to the rest of the already existing jobs. Note that after this process, a previously free job now has value.

Table 2: Job data fields.

Field	Description
id	to identify the job
job_files	location of the job files
hours	how long in standard units to run the job
money	total value of completing the job
results_files	location of the results files
taken	boolean if this job is being run currently



Job ID	Value	Hours
Dengue 1	$10 per hour	10
Dengue 2	$10 per hour	10
Malaria 1	$0 per hour	10
Malaria 2	$0 per hour	10
Malaria 3	$0 per hour	10

Job ID	Value	Hours
Dengue 1	$12 per hour	10
Dengue 2	$12 per hour	10
Tuberculosis 1	$10 per hour	20
Tuberculosis 2	$10 per hour	20
Malaria 1	$2 per hour	10
Malaria 2	$2 per hour	10
Malaria 3	$2 per hour	10

Figure 2: Example states of the database before and after two new jobs are submitted.

Next, an end user requests a job. The server determines which jobs the user can complete in the maximum allowed time by dividing the standard job time by the user's computer rating and comparing it to a constant maximum. From this list, the highest paying job is sent to the user, along with the appropriate files. At this point, the files are installed and run by the client, before sending back the results files generated. When the results files are sent back, they are stored with the job, and the end user is payed. At this point, the researcher can request their files and the job is deleted.

The central authority monitors users and researchers both at the time of registration and throughout the use of the system. Stringent checks are made for each researcher to avoid malicious behavior. The number of completed jobs for each end user, and the number of submitted jobs for each researcher, is tracked as a measure of trustworthiness.

2.2 Security

There are several security issues that must be considered when essentially outsourcing computations like these. If an adversary wants to obtain the payout of a job without computing the actual simulation tasks, he/she can simply submit wrong results to the central authority. This poses a problem since validation is equivalent to re-computation. At the moment, the central authority simply re-computes the simulation data since any client could be malicious and there is no way to validate the computations through the clients.

The results of these computations are currently only known by the client that executes a job and the researcher who requested the job, but there is no limitation that the client cannot send the results elsewhere. In the future, these computations may contain sensitive data, so there should be a mechanism to keep the results hidden from the client, which could be malicious. Encrypting the simulation inputs would effectively obscure the contents of the simulation, and the simulation operations could be performed under a partially homomorphic encryption scheme in order to preserve confidentiality on the intermediary results. This solution is extremely computationally intensive; a better solution might be to replace the simulation inputs with isomorphic compounds that have similar behavior without revealing the desired inputs.

The researchers that submit a job could be malicious as well – an adversarial researcher could submit a computationally infeasible job to burn clients' clock cycles. Moreover, the difficulty of a job may not be apparent from the simulation inputs. To resolve this issue, we would need to employ some heuristics on the program inputs, developed by a subject matter expert, to prevent a client from working on a job that is effectively impossible for them. Given the amount of protein folding and molecule binding simulations that run on high-performance computers, there should be a way to determine a ball-park operational complexity of tasks. The clients would then be able to adjust their jobs based on their hardware capabilities.

3 Modeling

3.1 Preprocessing

Before any real computation, the raw metadata of the Pathogen Box Database is filtered and a molecule string in Simplified Molecular Input Line Entry Specification (SMILES) format that is labeled with the Pathogen Box ID is extracted. SMILES strings are basic line input that include connectivity of atoms but no coordinated positions. Additionally, hydrogen atoms are not included. A simple example to consider would be Ethanol, C_2H_6O which is represented in SMILES format as CCO. For more complicated molecules, the equals sign represents double bonds, the pound sign represents single bonds, and a set of parentheses represents branches.

For this project, the SMILES format molecule is converted using an online Protein Data Bank (PDB) file creation tool from the National Cancer Institute [5]. The PDB format is a fixed width specification of atom coordinates in angstroms along connectivity bonds [6], but is also used for other molecules. The online tool automatically chooses an orientation and placement for the atoms, making guesses at the relative orientation of certain bonds. Multiple molecules can be specified using keywords. In order to adjust the PDB files for this project, we then replicate each drug molecule multiple times a specified number of angstroms apart in a cubic grid and clean up the PDB files using appropriate labels for each residue. The general flow for this process is summarized in Figure 3.

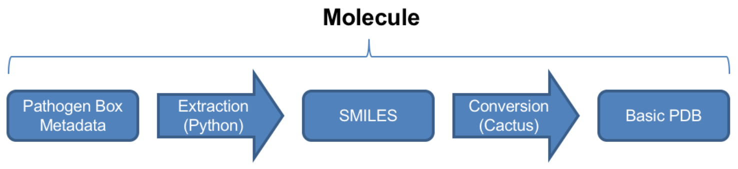


Figure 3: The data flow path for preparing a drug molecule.

Although many forcefield standard files are available for OpenMM to use with protein, DNA, and water simulations, they do not work with the PDB files generated from these drug moleucles. Instead we use the Amber tools package, which provides tools for molecular simulations in order to create an appropriate force field for simulation in the form of parameter/topology (prmtop) and coordinate (inpcrd) files. The prmtop format describes the topology of the molecule along with its bond forces and angles. The inpcrd format describes the x, y, and z coordinates for the molecule [3]. Using the reduce function, any missing hydrogen atoms are added to the PDB file, then using solvatebox and a standard water model (water.tip3p), we fill the remaining space with water molecules. Finally the antechamber function converts everything to the correct format for simulation. The general flow for this process is summarized in Figure 4.

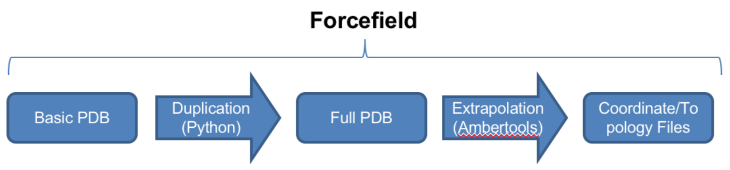


Figure 4: The data flow path for preparing the forcefield.

3.2 Simulation

In this section, we abstract the bounty system as a single computer that computes molecular dynamics simulations under our direction, and focus on a pipeline from the Pathogen Box Database to data about each molecule's behavior in water.

Each drug molecule in water solvent is then simulated using OpenMM, part of the Omnia Suite. OpenMM is the premiere open source tool for these molecular dynamics simulations; it is used by Folding@Home and Curecoin, has been optimized for GPUs and CPUs, and can be accessed using a variety of languages in techniques. For this project Python is used. Using the standard settings, we simulate the behavior of our systems over small timescales while recording atom positions and PDB information. This data can be used to determine if the molecules aggregate in water. The general flow for this process is summarized in Figure 5.

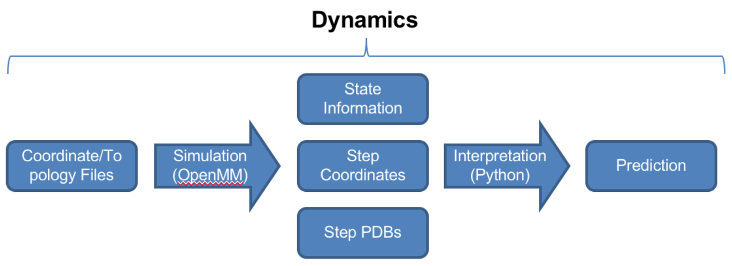


Figure 5: The data flow path for providing a prediction.

4 Results

4.1 Data

The data for this project is derived from information about the contents of The Pathogen Box from Medicines for Malaria Venture (MMV, Switzerland) [2]. It consists of 400 drug-like molecules active against a range of neglected subtropical diseases, as seen in Figure 6. The box of chemical compounds itself and the surrounding data are provided free of charge. We use the metadata fields ID, disease category, and SMILES structure for our work.

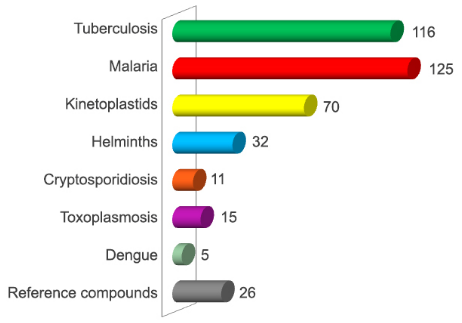


Figure 6: Distribution of molecules for each disease.

4.2 Aggregation

We have provided a system to generate detailed information at given timestamps about each of the 400 drug molecules under standard conditions in a small box of water. For our project, we use the data to evaluate whether the given molecule can be transmitted in blood, or whether it aggregates. Unfortunately, time constraints related to the steep learning curve and lengthy simulations prevented us from generating all of the data in enough time to evaluate the results. However, we are confident the prototype server-client system, when used by someone with more scientific expertise, could be used to generate results for any problem space.

In order to determine if a drug molecule coagulated in a given simulation, the radial distribution function is used. The radial distribution function characterizes the density of a system relative to an arbitrary specified particle, and is the probability of finding a particle a given distance from that reference particle. This is calculated by computing the distances between all pairs of particles and binning them into a histogram, as in figure 7. This histogram is typically normalized relative to an ideal gas [7].

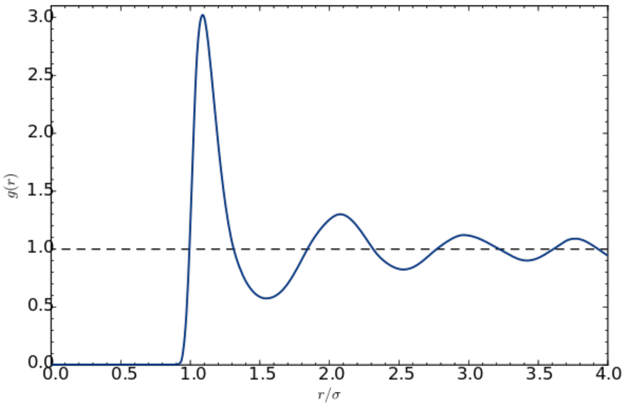


Figure 7: Density of a molecular fluid system.

For our system, we compute the histogram for the water molecules and the drug molecules separately. We expect that the water molecules will be distributed uniformly at a close distance, and will in fact have an initial peak similar to the histogram shown. If the drug molecule is uniformly distributed, its histogram will have peaks at a larger distance; however, if the drug molecule coagulates, its histogram will have a similar large peak at the start of the graph. We go further as well, computing histograms at each results time step for the system. As time progresses, we can observe the peak moving as the system coagulates, or remaining stable as the system stays uniformly distributed.

5 Conclusions

In this project, the objective was to design and create a biologically useful system that incorporate data mining techniques. We completed this goal by simulating drug molecules in water to predict aggregation, using a distributed client-server system for the simulations. During the process we gained experience with tools in the molecular dynamics area, learned a variety of chemical concepts, and practiced designing and implementing an application.

There are a variety of directions for future work on this project. For this specific application of molecular dynamics, we would first and foremost like to deliver the predictions for the Pathogen Box drug molecules, preferably with more appropriate standards and settings. We would also suggest using a more descriptive basepoint for the molecule rather than the SMILES string. On the server side, currently the system runs on a single computer, and needs a dedicated platform set up that is accessible using the clients. Although research has gone into designing security solutions, the more complicated approaches still need to be implemented. Most importantly, new job types should be added beyond molecular dynamics simulations in order to test the generalized approach we have taken.

Acknowledgments

Many thanks to Dr. Arvind Ramanathan, Dr. Scott Emrich, and especially Dr. RJ Nowling for their expertise in selecting and developing a problem space for mining.

References

[1] About. `http://folding.stanford.edu/about/`. Accessed: 2018-02-27.

[2] About the pathogen box. `https://www.pathogenbox.org/about-pathogen-box`. Accessed: 2018-02-27.

[3] Amber file formats. `http://ambermd.org/formats.html`. Accessed: 2018-03-20.

[4] Frequently asked questions. `https://curecoin.net/knowledge-base/knowledge-base-faq/`. Accessed: 2018-02-27.

[5] Online smiles translator and structure file generator. `https://cactus.nci.nih.gov/translate/`. Accessed: 2018-02-27.

[6] Protein data bank contents guide. `ftp://ftp.wwpdb.org/pub/pdb/doc/format_descriptions/Format_v33_Letter.pdf`. Accessed: 2018-03-20.

[7] David Chandler. *Introduction to modern statistical mechanics*. New York: Oxford University Press, 1987.

[8] David E. Shaw; Martin M. Deneroff; Ron O. Dror; Jeffrey S. Kuskin; Richard H. Larson; John K. Salmon; Cliff Young; Brannon Batson; Kevin J. Bowers; Jack C. Chao; Michael P. Eastwood; Joseph Gagliardo; J.P. Grossman; C. Richard Ho; Douglas J. Ierardi; Istvn Kolossvry; John L. Klepeis; Timothy Layman; Christine McLeavey; Mark A. Moraes; Rolf Mueller; Edward C. Priest; Yibing Shan; Jochen Spengler; Michael Theobald; Brian Towles; Stanley C. Wang. Anton, a special-purpose machine for molecular dynamics simulation. *Communications of the ACM*, 51(7):9197, July 2008.

6 Appendix

`https://github.com/ParkerDiamond/OpenMM-PubSub`