Simulation of Molecular Dynamics Using OpenMP

Ayushman Khuntia
B.Tech (CSE)
Vellore Institute Of Technology
Vellore, India
khuntia.ayushman@gmail.com

Pushpa Gothwal
Professor
Vellore Institute Of Technology
Vellore, India
pushpa.gothwal@vit.ac.in

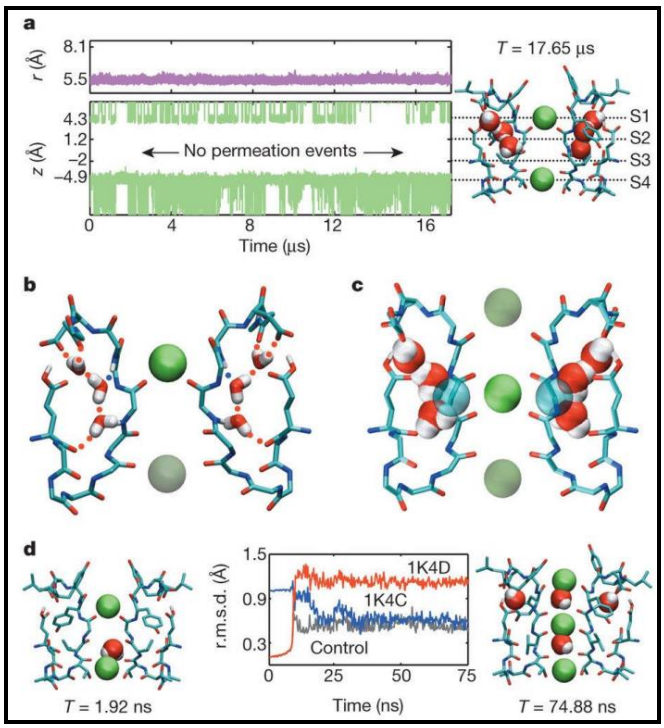
***Abstract*— Molecular-dynamics (MD) simulation is a well-progressed numerical method that shows the use of a suitable algorithm to solve the classical equations of motion for atoms relating with a known inter atomic potential. Techniques for MD simulation are also ideal for researching surface phenomena because they provide a qualitative understanding of surface dynamics and structure. In order to save time, we shall parallelize the serial simulation of molecular dynamics.**

Keywords—Molecular Dynamics, simulation, parallelize, inter atomic potential

I. INTRODUCTION

For many years now, molecular dynamics has been utilised to comprehend how the dynamics of liquids, solids, and liquid-solid interfaces vary with temperature and pressure. Since they provide a qualitative grasp of surface structure and dynamics, MD simulation approaches are also well suited for comprehending surface phenomena. In order to understand surface disorder and pre melting better, this project employs MD techniques. In general, it investigates how the structure and vibration dynamics at the surfaces of face centered cubic (FCC) metals—primarily Ag, Cu, and Ni—are affected by temperature. Moreover, results from different theoretical and experimental techniques are exchanged. The MD approach has been used on a broad range of surfaces, including those of semiconductors, insulators, alloys, glasses, and simple or binary liquids, even though the focus of this project is on metal surfaces. A thorough analysis of the benefits and drawbacks of the approach as it relates to these really intriguing systems is beyond the purview of this Study. It is crucial to note that the accuracy with which the forces acting on the ion cores can be calculated determines whether the conventional MD simulation will be successful. The success of molecular dynamics simulations has made them more suitable for such designs than traditional MD simulations on semiconductor and insulator surfaces. With the use of a computer, we can run MD Simulation on thousands of atoms and determine the structure of one atom based on its interactions with other atoms. This article describes GROMACS, one of the MD programmes created specifically for proteins. The results of this investigation convinced us that every protein has a unique half-life. We can comprehend how proteins fold and unfold, thanks to Gromacs.

Large-scale simulation is expected to play a bigger role in the future as a result of the success of large-scale molecular dynamics applications. The gap between atomic level attributes and entire cell activity is being bridged through mutual simulation and experimentation, a task that cannot be accomplished by either strategy alone.



Source : Britannica

II. LITERATURE REVIEW

Previous works have explored the simulation of molecular dynamics using various technologies. Below is a detailed survey of the literature on this topic:

"Using FPGA devices to accelerate biomolecular simulations" by **S. Alam, P. Agarwal, M. Smith, J. Vetter, D. Caligaet et al (2007)** – They used Field programmable gate array devices and Verlet integration method of parallelism. Its limitations include the following :

Cost: FPGA devices can be expensive compared to traditional computer hardware, and the cost of the hardware and development tools can be a barrier to wider adoption.

Complexity: The design and implementation of algorithms for FPGAs can be complex and time-consuming, and requires specialized skills and expertise.

Flexibility: FPGA devices are less flexible compared to traditional computer hardware, as they are designed for specific tasks and applications. This can make it difficult to adapt the hardware to new or evolving requirements.

Limited scalability: The performance of FPGA-based biomolecular simulations may be limited by the number of processing elements available on the FPGA device. This can limit the size and complexity of the simulations that can be performed.

"Quantum Monte Carlo on graphical processing units" by **A.G. Anderson, W.A. Goddard, P. Schroder et al (2007)** – They used the Metropolis algorithm and parallel processing capabilities of GPU over CPU. The limitations of this paper were following :

GPU hardware limitations: The performance of GPU-based QMC simulations can be limited by the hardware specifications of the GPU, such as memory size, processing power, and bandwidth.

Data transfer: The data transfer between the GPU and the CPU can be a bottleneck for GPU-based QMC simulations, and can result in performance degradation and increased latency.

Energy consumption: GPUs consume significant amounts of energy, and running large-scale QMC simulations on GPUs can result in high energy consumption, which can be a concern for data centers and high-performance computing environments.

Programming complexity: The programming of GPU-based algorithms can be complex, and requires specialized skills and expertise in GPU programming.

GPU-specific algorithms: The algorithms presented in the paper are specifically optimized for GPUs, and may not be suitable for other types of hardware, such as CPUs or Field Programmable Gate Array (FPGA) devices.

"Molecular-Dynamics Simulations using Spatial Decomposition and Task-Based Parallelism" by **Mangiardi C.M et al (2016)** - This study investigates the algorithm of spatial decomposed parallelism and task based parallelism. The limitations include : Accuracy, scalability, Load balancing, Synchronisation.

"Introduction to molecular dynamics simulation" by **MP Allen, et al**. **[4]** (2004) - This study uses Verlet algorithm, Nose – Hoover algorithm and SIMD vectorization. Apart from being slow, the limitations include :

Age: The paper was published in 2004, and some advancements in the field may not be covered.

Focus on basic algorithms: The paper focuses on the basic algorithms used in molecular dynamics simulations, and more advanced topics such as enhanced sampling methods and multiscale simulations may not be covered in detail.

Limitations of molecular dynamics: The paper focuses on molecular dynamics simulations and their limitations, such as their reliance on empirical potentials and the difficulty in capturing quantum mechanical effects.

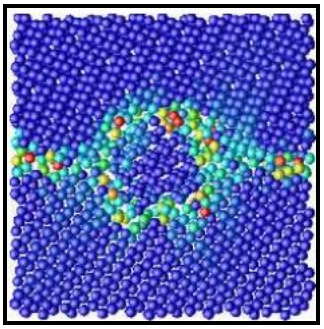
Lack of specific applications: The paper provides a general overview of molecular dynamics simulations, but does not provide specific examples of applications to particular systems or problems.

III. KEY PROPERTIES OF MOLECULAR DYNAMICS

A. Atoms Never Stop Jiggling

In real life, and in an MD simulation, atoms are in constant motion. They will not proceed to an energy minimum and keep there. Given enough time, the simulation examples the Boltzmann distribution.

- That is, the probability of detecting a specific arrangement of atoms is a function of the potential energy.
- In reality, one often does not simulate long enough to reach all energetically favorable arrangements.
- This is not the only way to explore the energy surface (i.e., sample the Boltzmann distribution), but it's a pretty effective way to do so.



Source :Wikipedia

B. Energy Conservation

Total energy (potential + kinetic) should be conserved.

- In atomic arrangements with lower potential energy, atoms move faster.
- In practice, total energy inclines to grow gradually with time due to numerical errors (passing errors).
- In a lot of simulations, one adds a mechanism to keep the temperature unevenly constant (a "thermostat")

C. Water Is Important

Overlooking the solvent (the molecules nearby the molecule of interest) leads to main artifacts examples are Water, salt ions (e.g., sodium, chloride), lipids of the cell membrane. Two choices for taking solvent into account – Plainly denote solvent molecules.

- High computational cost but more precise.
- Usually accept periodic boundary settings (a water molecule that drives off the left side of the simulation box will come back in the right side, like in PacMan).
- Implicit solvent
- Mathematical model to imprecise average effects of solvent.
- Less precise but faster

IV. LIMITATIONS OF MOLECULAR DYNAMICS SIMULATION

A. Time Scale

Simulations need short time stages for numerical stability. 1 time step ≈ 2 fs (2×10–15 s). Structural changes in proteins can take nanoseconds (10–9 s), microseconds (10–6 s), milliseconds (10–3 s), or longer – Millions to trillions of sequential time steps for nanosecond to millisecond events (and even more for slower ones) Until lately, simulations of 1 microsecond were unusual. Advances in computer power have allowed microsecond simulations, but simulation

periods continue a challenge. Enabling longer-timescale simulations is an active research area, connecting :

- Algorithmic Developments
- Parallel Computing
- Hardware : GPUs, specialized hardware

B. *Force Field Accuracy*

Molecular mechanics force fields are integrally calculations. They have improved substantially over the last 10 years, but many boundaries still remains. In repetition, one needs some experience to know what to trust in a simulation.

C. *Covalent Bonds cannot break or form during (standard) MD Simulations*

Once a protein is formed, most of its covalent bonds don't break or form during characteristic function. A few covalent bonds form and break more normally (in real life) :

- Disulfide bonds between cysteines
- Acidic or basic amino acid residues can lose or gain a hydrogen (i.e., a proton)

V. METHODS TO SPEEDUP MD SIMULATIONS

A. *Reduce the amount of computation per time step*

Faster algorithms. Example: fast estimated methods to compute electrostatic interactions, or methods that allow you to assess some force field terms every other time step.

B. *Reduce the number of time steps required to simulate a certain amount of physical time*

One can increase the time step several fold by slowing out some very fast motions (e.g., certain bond lengths).

C. *Reduce the amount of physical time that must be simulated*

A major research area includes making events of interest take place more quickly in simulation, or making the simulation reach all low-energy conformational states more quickly. For example, one might implement artificial forces to pull a drug molecule off a protein, or push the simulation away from states it has already stayed. Each of these methods is active in certain specific cases.

D. *Parallelize the simulation across multiple computers*

Splitting the computation associated with a single time step across multiple processors needs communication between processors. In our project we are about to use this method.

- Generally each processor takes accountability for atoms in one spatial region.
- Algorithmic improvements can reduce communication necessities.
- Perform many short simulations. One research goal is to use short simulations to predict what would have happened in a longer simulation.

E. *Redesign computer chips to make this computation run faster*

GPUs (graphics processor units) are now extensively used for MD simulations. They pack more arithmetic logic on a chip than traditional CPUs, and give a substantial speedup. Parallelizing through multiple GPUs is difficult. Several projects have built chips especially for MD simulation. These pack even more arithmetic logic onto a chip, and allow for parallelization through many chips.

VI. PROPOSED WORK

We will write a code which will expect the kinetic and potential energy of an atom at all time step which will help us in foreseeing the motions (velocity, position acceleration etc) of the atoms. The sum of kinetic energy and potential energy must be constant and equal to total energy our code will also print the error in total energy which will tell us about what percentage of error we can get in our calculations.
Below we have given the problem analysis chart (PAC) of our simulation code for better accepting of the MD Simulation.

TABLE I. PAC (PROBLEM ANALYSIS CHART)

Input	Processing	Output	Solution Alternative
Spatial Dimension No. of particles in Simulation No. of Time Steps Size of Each Time step	At every step, our simulation should report the potential and kinetic energies. The sum of these energies should be a constant. As an accuracy check, our simulation should also print the relative error in the total energy. We should parallelize the code and associate the time taken in both serial and parallel execution.	Relative Error in Total energy. Time difference between Serial and parallel Execution.	NONE

The main outcome we are expecting from this project is the decrease in amount of time taken to compute the potential and kinetic energies of atoms and that is also the reason why we are paralleling the compute and update function as they do most of the calculation. Another Outcome which we are expecting from this simulation is decrease in the error of theoretical and practically observed energies. Lesser the error, more the accuracy but we also know we cannot have the error free output but we are doing our best to make it very small by do not taking approximation of most of the values in calculation part.

Functions in our application code :

Compute: - COMPUTE computes the forces and energies of atoms.

Initialize: - INITIALIZE initializes the positions, velocities, and accelerations.

Timestamp: - TIMESTAMP prints the current YMDHMS date as a time stamp.

Update: - UPDATE updates positions, velocities and accelerations.

Displacement: - DISPLACEMENT computes the displacement between two particles.

CPU Time: - CPU_TIME reports the elapsed CPU time.

Compute and update function are the most time overriding functions as they are only about computation and solving mathematical equations so we are going to parallelyze both of them. Some parallel function which we are about to use in our code are pragma omp parallel, pragma omp reduction, pragma omp for, pragma omp shared, pragma omp private. The major extensive thing in our project will be the time difference in the serial code and parallel code because the main goal of our project is to decrease the time taken in molecular dynamics simulation.

In mathematics, numerical analysis, and numerical partial differential equations, domain decay methods solve a edge value problem by splitting it into smaller boundary value problems on sub domains and repeating to coordinate the solution between adjacent sub domains. A coarse problem with one or few unknowns per sub domain is used to further organize the solution between the sub domains globally. The problems on the sub domains are independent, which makes domain decay methods suitable for parallel calculating.

VII. CALCULATIONS INVOLVED

The Basic algorithm is we divide time into discrete time steps, no more than a few femtoseconds (10–15 s) each and at each time step we calculate the forces performing on each atom, using a molecular mechanics force field and when the drive in atom occurs we update position and velocity of each atom using Newton's laws of motion.

We have simple equations of motion and used them to do the computations.

Newton's second law: F = ma – where F is force on an atom, m is mass of the atom, and a is the atom's acceleration Recall that: – where x represents coordinates of all atoms, and U is the potential energy function.

Velocity is the derivative of position, and acceleration is the derivative of velocity. We can thus write the equations of motion as:

$$F(x) = -\nabla U(x)$$

$$dx/dt = v \text{ and } dv/dt = F(x)/m.$$

This is a system of ordinary differential equations – For n atoms, we have 3n position coordinates and 3n velocity coordinates. "Analytical" (algebraic) solution is difficult but Numerical solution is straight forward.

$$xi + 1 = xi + \delta t\ vi$$

$$vi + 1 = vi + \delta t\ F(xi)/m$$

Above is the numerical open solution where δ t is the time step.

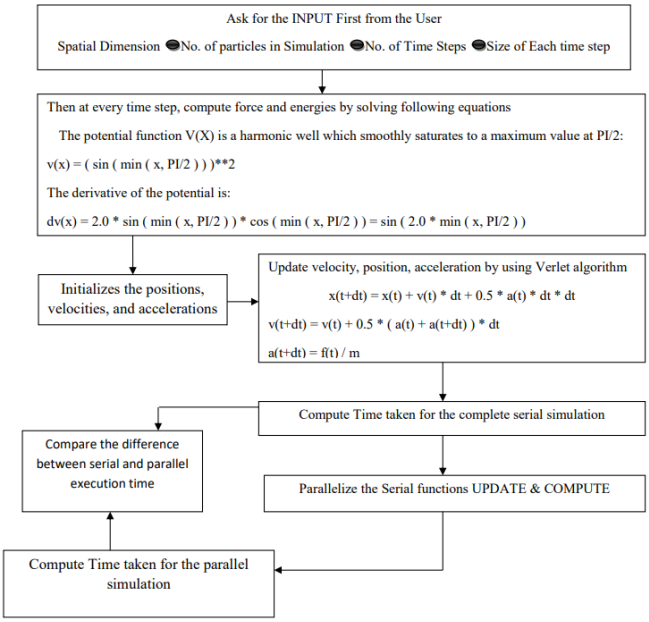
In practice, people use "time symmetric" integration methods such as "Leapfrog Verlet"

$$xi + 1 = xi + \delta t\ v(i+1)/2$$

$$v(i + 1)/2 = v(i-1)/2 + \delta t\ F(xi)/m$$

This gives more correctness.

TABLE II. FLOW CHART OF IMPLEMENTATION



Calculating the interactions : As a result of using periodic limit conditions, each particle is in the simulation box appears to be relating not only with the other particles in the box, but also with their images. Actually, the number of interacting pairs increases extremely. Nevertheless, this problem can be overcome if one uses a potential with a finite range, since the communication of two particles separated by a distance exceeding a cut off distance can be overlooked. Assuming that the box size is larger than 2Rcut along each Cartesian direction, it is clear that at most one among the pairs made by a particle i in the box and all the periodic images of another particle j will lie inside the cut off distance Rcut and, thus, will interact. This is the kernel of the minimum image principle: among all images of a particle, study only the closest and neglect the rest. **Correcting particle coordinates** : After each integration step the coordinates of the particles must be studied. If a particle has left the simulation region, its coordinates must be readjusted to bring it back inside, which is conforming to bring in an image particle through the opposite boundary. Supposing that the simulation region is a rectangular box, this is done by adding to or subtracting from the affected particle coordinate the size L of the box along the conforming direction. Thus, for instance, supposing that the x-coordinates are defined to lie between 0 and L, if $x < 0$ (the particle leaves the box in the negative Ox direction), then the coordinate must be corrected by adding the box size L. If $x > L$ ˙ (the particle exits in the positive Ox direction), L must be subtracted from the particle coordinate:

$$\text{if } x < 0 \text{ then } x \rightarrow x + L$$

$$\text{if } x > L \text{ ˙ then } x \rightarrow x - L.$$

An alike analysis must be carried out for each coordinate of the particle. The readjustment of the coordinates in the context of using periodic limit conditions has been applied in the following routine.

VIII. COMPARISON

TABLE III. Proposed Method Metrics

S. No.	*Time Step*	*Difference in Serial and parallel computation time*
1.	300	7.962 s
2.	1000	26.862 s
3.	2000	53.950 s

TABLE IV. Existing Method Metrics

S. No.	*Time Step*	*Difference in Serial and parallel computation time*
1.	300	12.231 s
2.	1000	44.665 s
3.	2000	67.491 s

Note : Proposed system was executed on 2 cores and 2 threads and base system was executed on 3 cores and 3 threads.

IX. CONCLUSION

Molecular dynamics is an significant computational tool to simulate and comprehend biochemical processes at the atomic level. However, precise simulation of processes such as protein folding needs a large number of both atoms and time steps. This in turn leads to huge runtime requirements. Hence, looking for fast solutions is of highest importance to research.

In this project we present a new approach to accelerate molecular dynamics simulations with inexpensive product graphics hardware. To derive an effective mapping onto this type of computer architecture, we have used the new Compute Unified Device Architecture programming interface to implement a new parallel algorithm.

Our experimental results show that the parallel algorithm based method allows speedups of up to fifteen seconds compared to the corresponding sequential implementation. This speed up can also be enhanced when we increase number of atoms and number of time steps.

The energy or force calculation is the most time using part of almost all Molecular dynamics Simulation. If we take a model system with pair wise additive interaction (as done in many molecular simulation), we have consider the contribution to the forces on the particle I by all it neighbor. If we do not shorten the interaction, this implies that, for a system of N particles, we must evaluate N(N-1)/2 pair interaction.

Even if we do shorten the potential, we still would have to compute all N(N-1)/2 pair distances to define which pairs can interact. This implies that, if we use no tricks, the time needed for assessment of the energy scales as N2 . There exit efficient techniques for speeding up the assessment of both short-term and long term contact in such a way that the computing time scales as N3/2, rather than N2 . The technique which we have used is combination of Verlet and cell lists.

The first question that rises is when to use which method. This depends mostly on the details of the system. In any event, we always start with a arrangement as simple as possible, hence no trick at all. Although the algorithm scales as N2 , it is straightforward to implement and thus the probability of programming errors id relatively small. In addition we should consider how often the program will be used.

The use of verlet list become beneficial if the number of particles in the list is significantly less than the total number of particles in three dimension this means

$$nv = (4/3)*\pi r3\ \rho << N$$

After completing our project finally we can accomplish that writing code in OpenMP for molecular dynamics simulation helped in decreasing time of evaluation of forces or energies which was our main purpose when we choose this topic as our project and we are happy that we have successfully reduced time period by 15 seconds

X. FUTURE SCOPE

We can include Graphics processing units (GPUs), originally developed for interpreting real-time effects in computer games, now provide unparalleled computational power for scientific applications. We can develop a general purpose molecular dynamics code that runs completely on a single GPU. It is expected that our GPU application can provide a presentation equivalent to that of fast 30 processor core distributed memory cluster. Results of this can show that GPUs implementation as an inexpensive another to such clusters and discuss implications for the future.

Currently, the peak presentation of state-of-the-art GPUs is roughly ten times faster than that of comparable CPUs. Furthermore, the growth rate of the number of transistors used on GPUs is greater than for microprocessors. Therefore, GPUs will become an even more useful alternative for high performance calculating in the near future. And these facts aim and excite us to apply molecular dynamics using GPU.

The Compute Unified Device Architecture (CUDA) is a new hardware and software architecture for delivering and managing computations on GPUs. It treats the GPU as a data-parallel computing device without the need of mapping calculations to the graphics pipeline. By using the standard C language, CUDA makes it easy GPU-based software development for scientific calculating compared to previously used graphics-oriented languages such as OpenGL or Cg.

Molecular dynamics (MD) is a computationally concentrated method of studying the natural time-evolution of a system of atoms using Newton's classical equations of motion. In practice, MD has always been limited more by the current available computing power than by investigators' resourcefulness. Researchers in this field have typically concentrated their efforts on simplifying models and finding what may be neglected while still obtaining acceptable results. MD simulations can benefit from the calculating power of GPUs. In order to exploit the GPU's capabilities for high performance MD simulation, we present a new algorithm for non-bonded short-range connections within the atom system. The algorithm has been applied using C++

and CUDA and tested on a physical system of up to 131,072 atoms. We show that our new MD algorithm leads to a performance development of one order of scale on a product NVIDIA GeForce 8800 GTX card.

REFERENCES

[1] S. Alam, P. Agarwal, M. Smith, J. Vetter, D. Caliga "Using FPGA devices to accelerate biomolecular simulations" Computer, 40 (3) (2007), pp. 66-73

[2] M.P. Allen, Introduction to molecular dynamics simulation, in: Computational Soft Matter—From Synthetic Polymers to Proteins, NIC Series, vol. 23, John von Neumann Institute for Computing, 2004, pp. 1–28

[3] A.G. Anderson, W.A. Goddard, P. Schroder Quantum Monte Carlo on graphical processing units Computer Physics Communications, 177 (2007), pp. 298-306

[4] J.A. Anderson, C.D. Lorenz, A. Travesset General purpose molecular dynamics simulations fully implemented on graphics processing units, Journal of Computational Physics, 227 (2008), pp. 5342-5359

[5] R.G. Belleman, J. Bédorf, S.F. Portegies Zwart High performance direct gravitational N-body simulations on graphics processing units II: An implementation in CUDA, New Astronomy, 13 (2008), pp. 103-112

[6] Mangiardi C.M. (Master's thesis), Molecular-Dynamics Simulations using Spatial Decomposition and Task-Based Parallelism, Laurentian University, Sudbury, Ontario (2016)

[7] Pal A., Agarwala A., Raha S., Bhattacharya B., J. Parallel Distrib. Comput., 74 (2014), pp. 2203-2214

[8] Zhiwei Zhang; Pei Chen; Fei Qin, "Molecular dynamics simulation on subsurface damage layer during nano grinding process of silicon wafer", 18th International Conference on Electronic Packaging Technology (ICEPT) pp. 487-490,2017

[9] Soni S, Tyagi C, Grover A1 , Goswami SK, "Molecular modeling and molecular dynamics simulations based structural analysis of the SG2NA protein variants" ,NCBI ,2014.

[10] M.P. Allen, D. Frenkel, J. Talbot Molecular dynamics simulation using hard particles , Comput. Phys. Rep., 9 (1989), pp. 301-353

[11] D. Mandal, A. Shukla, A. Ghosh, A. Gupta and D. Dhabliya, "Molecular Dynamics Simulation for Serial and Parallel Computation Using Leaf Frog Algorithm," 2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan, Himachal Pradesh, India, 2022, pp. 552-557, doi: 10.1109/PDGC56933.2022.10053161.