Monte Carlo Simulation of the Energetics and Thermodynamics of Polypyrrole in Condensed Phases

Greg Helmick[a,b], Yoseph Abere[a,b] and Estela Blaisten-Barojas[a,b,*]

[a]Center for Simulation and Modeling,
[b]Department of Computational and Data Sciences,
George Mason University, Fairfax, VA 22030, USA, blaisten@gmu.edu

ABSTRACT

Polypyrrole (PPy) is a conducting polymer with a wide range of applications such as super-capacitors, sensors, batteries, actuators, and neural prosthetics. PPy is a popular polymer in experimentation not so easily accessed in computational modeling efforts. As such modeling and algorithmic improvements are crucial in its study. Presented here are both a novel coarse grain model for oxidized PPy and its efficient and scalable computing implementation of a Monte Carlo study of the energetics and thermodynamics of the PPy condensed phases. Systems containing 10^3 to 10^5 particles have been meticulously implemented in a range of computer platforms ranging from desktop class computing hardware to high performance clusters equipped with multiple GPUs. Our implementation utilizes a combination of CPUs and GPUs depending on system size. Reported properties are the density, enthalpy, cohesive energy, compressibility, thermal expansivity, Hildebrand solubility parameter, bulk modulus, and pair correlation functions at ambient conditions. All properties are consistent across the studied system sizes and in good agreement with experimental values if available.

Keywords: polypyrrole, oxidized PPy, Monte Carlo, GPU computing, PPy force field

1 INTRODUCION

Organic metals, also termed conducting polymers, have promoted a host of novel applications [1]. In parallel, more efficient cyber methods were necessary to predict the thermodynamic, mechanical and electronic properties of the new polymeric materials. For example, the electronic structure of molecular crystals is now obtained through Density Functional Theory (DFT), a faster computational approximation than *ab initio* methods. However, DFT is limited to a relatively small number of atoms typically less than 1,000 [2]. To reach more realistic systems containing more atoms, classical approaches with parameters based on DFT have increased in the modeling world. Model potentials are the mathematical description of how the particles that compose the system interact, describing a system atomistically or coarse grained (CG) by including groups of atoms as individual entities. CG models permit system size scaling to significantly larger systems [3]. High performance computing (HPC) methodologies such as Molecular Dynamic (MD) and Monte Carlo (MC) combined with CG modeling are powerful tools for studying the mechanical and thermodynamical properties of very large systems of particles.

Algorithmic improvements are crucial in all HPC areas, whether targeting CPU, GPU or a combination of the two [4, 5]. Indeed, GPU-GPU processing has become a staple in HPC, both in MC [6] and in MD [7] implementations. These hardware and algorithmic advances have permitted the simulation of systems of increasing complexity, size and temporal evolution. In this brief we present a battery of Metropolis Monte Carlo (MMC) optimized implementations for a family of novel, in-house, CG model potentials of oxidized polypyrrole (PPy). This conducting polymer is used in super-capacitors, sensors, batteries, actuators, neural prosthetics, among other applications. PPy is easy to synthesize, as well as being low cost; these attributes make PPy a popular and useful polymer [8].

2 MODEL AND METHODS

The CG model potential for PPy oligomers containing 12 monomers, termed 12-Py, contains halogen atoms as dopants. The oligomer monomers are modeled as a united planar group of atoms with one dipole moment anchored at its center. When doping occurs, there is charge transfer between the 12-Py chains and the dopants. As a result, the oligomers become cations with a +4 charge and the dopants acquire a -1 charge. The ratio dopant/oligomer in the mixture considered is 4/1, which is consistent with the experimental syntheses. Figure 1 gives a schematic representation of the 12-Py oligomer in an environment of dopants.

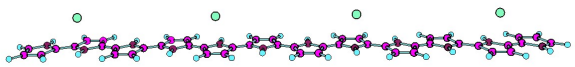

Figure 1: Schematic view of the 12-Py oligomer with four atomic dopants arbitrarily positioned close to the polymer chain.

The model potential has two main contributions, in-

ternal to the 12-Py oligomers (U_{intra}) and external to the polymer (U_{inter}) containing the interactions between oligomers U_{py-py}, oligomers and dopants U_{py-dop}, and between dopants $U_{dop-dop}$ [9]. Thus,

$$U_{total} = U_{intra} + U_{inter} \tag{1}$$

$$U_{intra} = U_{bond} + U_{bend} + U_{torsion} + U_{lib} + U_{elec} + U_{dip-dip} + U_{anti-coil} \tag{2}$$

$$U_{inter} = U_{py-py} + U_{py-dop} + U_{dop-dop} \tag{3}$$

Here, U_{bond} identifies the bonding between contiguous monomers within an oligomer, U_{bend} is the angular motion between two contiguous bonds, $U_{torsion}$ identifies changes in the angle between the plane of two contiguous monomers, U_{lib} pertains to librations within the monomer plane, U_{elec} is the Coulomb interaction, $U_{dip-dip}$ is the interaction between the monomer dipoles within the oligomer, and $U_{anti-coil}$ eliminates possible coiling of the oligomer chain. Meanwhile, the three terms in U_{inter} contain sums of electrostatic and dispersion interactions. When the interactions within each of the three terms in U_{inter} are summed, Eq. 3 reduces to sum of three pairwise additive functions visualized in Fig. 2. The parameters for this CG force field for PPy doped with chlorine atoms were reported in [9].

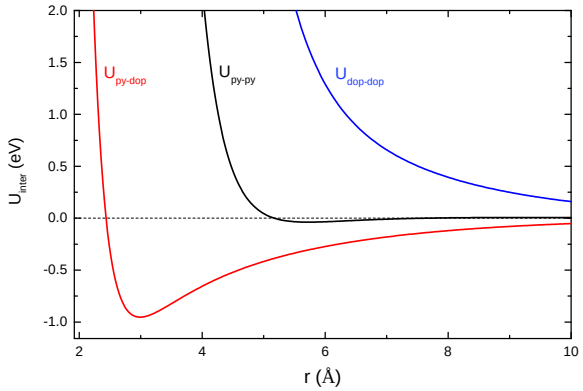

Figure 2: The three pairwise potentials that compose the U_{inter} contribution to the CG force field of PPy. Parameters used for the plot depiction correspond to PPy with chlorine dopants [9].

The term *particle* will be used herein to identify either each monomer of a 12-Py oligomer or a dopant atom. Systems ranging in size from 1024 to 27648 particles were explored for calibrating the size scaling effect on the computational implementation. Simulations were performed with custom-developed horizontally scalable GPU accelerated Metropolis Monte Carlo (MMC) implementation and supporting four simulation modes: NVT (canonical ensemble), NPT (isobaric-isothermal ensemble), plus NVT and NPT Adaptive Tempering MC [10].

The MMC algorithm is inherently serial since the next step in an MMC simulation depends on the current step. Efforts of the research were targeting on the parallelization as applied to particles interacting through the model potential as described previously [9]. Parallelization of particle simulations can typically be classified as one of three types: particle decomposition, domain decomposition, and farm or energy decomposition. The domain decomposition method utilizes a spatial decomposition approach that allows for simultaneous updates of particles within domains. Simultaneous movement of particles is accomplished by moving particles that are outside each other's cutoff radius. The energy decomposition method achieves parallelization by splitting the particles into groupings and then calculating partial energy values for each group. The partial values are combined to determine the total system energy. The parallelization method utilized here was a custom-modified energy decomposition that includes preliminary elements of our previous MMC parallel implementations [11, 12].

The energy decomposition applied to systems of doped PPy modeled as described above becomes computationally demanding as the system size increases. Hence, our performance optimization targeted specifically an efficient calculation of the three pair-additive interactions included in U_{inter} and depicted in Fig. 2. The first performance optimization involved multiple combination trials leveraging three computing parallelization components when implemented for a computer cluster with multiple nodes: Open MPI for the distributed memory in a cluster, OpenMP effective within one cluster node for using the intra-node shared memory, and CUDA for the GPU in each node. The combined approach of the three-level parallelization scheme was scalable to any sized computing platform.

The second optimization step focused on the calculation of the various terms of the model potential. Instead of the actual analytical expressions of the terms entering in U_{inter} (Eq. 3), a numerical approximation was implemented consisting of interpolating between precalculated values of the functions in Fig. 2, which were stored as a multidimensional table. This approximation resulted in a large execution time improvement. Polynomial interpolation was adopted due to the quick evaluation. The several different piecewise interpolation methods implemented were piece-wise linear, Hermite polynomials and natural splines. Hermite polynomials and natural splines yielded comparable root mean square errors (RMSE) of the density, potential energy and enthalpy. These interpolation methods were custom implemented to optimize the execution time on both the CPU or GPU. Natural splines was adopted for produc-

tion since the associated data tables are less cumbersome when the piece-wise interpolation interval between points of 0.001 a_o (a_o=Bohr radius) was used. Smaller intervals resulted in increased RMSE of the studied properties. Once determined, the coefficients of the involved polynomials were stored in a database, which was loaded to both the CPU and GPU memory prior to the simulation start.

3 RESULTS

As applied to the energy partition, the parallelization strategy is accomplished by partitioning the summations of the model potential terms over particles into sum segments, with each sum segment being assigned to a computational node. Hence, each node evaluates only a portion of the overall energies associated with the sum segment it was assigned. Within each node, the combination of OpenMP and CUDA provides further advantage in the distribution of the computational workload. The parallelization performance using the customized energy decomposition approach is shown in Fig. 3 illustrating the processing time of a single MMC passage over the particles when adopting different loads to the OpenMP (threads), Open MPI (processors), or GPU. As evidenced by the the Fig. 3 inset, for a number of particles below 10,000, the selection of how to combine processor and thread components is important because the CPU time may be doubled if care is not taken. As the system is scaled up in size to many more particles, the balance processor/thread is still important but not as damaging. Also evident from Fig. 3 is that the GPU use gave the best processing time performance. We emphasize that the performance metrics shown are for only one MMC step. In a MMC simulation for obtaining the physical properties of the PPy system, several millions of these steps are needed. It is clear that the usage of GPU acceleration for the adopted energy partitioning has an excellent size scalability. This effort utilized the PPy doped with chlorine atoms.

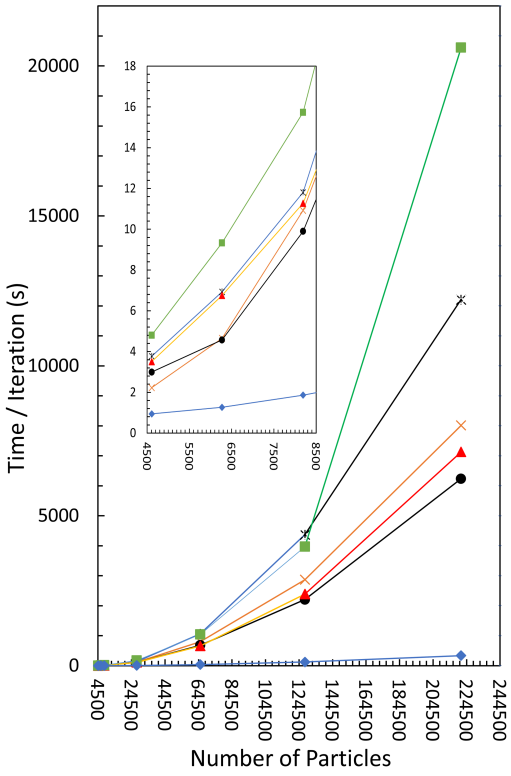

Figure 3: Comparison of MMC processing time per iteration between GPU and CPU with varying number of threads versus system size. Depicted are 1 process/32 threads (black asterisks), 4 process/8 threads (orange asterisks), 8 process/4 threads (black dots), 16 process/2 threads (red triangles), 32 process/i thread (green squares), 1 process/GPU (blue diamonds). The system was PPy doped with Cl atoms [9].

When using multiple nodes in a computer cluster via openMPI, the workload distribution described previously would occur per node. Hence, in a multi-node scenario each node is responsible for the evaluation of only an assigned portion of the U_{inter}, and if a GPU is available, the node will off-load the U_{inter} evaluation to the GPU.

The substitution of analytical expressions in Eq. 3 by a table of pre-calculated values with splines interpolation yielded significant performance gains, permitting the analysis of systems as large as 128,000 particles with the ability to scale up to even larger system sizes. Figure 4 evidences the achieved performance improvements

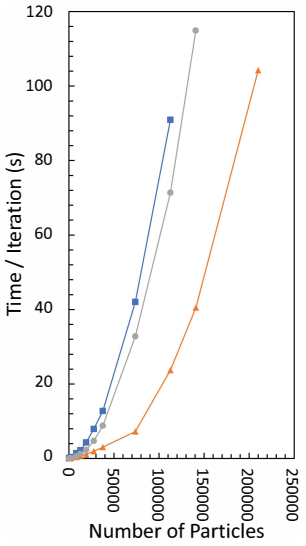

Figure 4: MMC processing time per iteration versus system size with U_{inter} calculated from a table of interpolated values. Implementation in CPU (gray) and in GPU (orange). For comparison, the GPU processing time using the analytical description of U_{inter} is colored blue.

obtained when using the two parallelization scenarios, either CPU alone or combined with GPU as compared with the best performance depicted in Fig. 3 for GPU implementation using the analytic description of U_{inter}.

Preliminary efforts for investigating PPy doped by fluorine and other halogen atoms are ongoing. Change of dopant in these polymeric systems implies different parameters entering in the model potential associated to U_{inter} that we are producing based on DFT calculations. In particular for fluorine, due to chlorine being a heavier atom, the density of fluorine doped systems would be about 93% the density of the chlorine doped system that we have reported as 1410 kg/m^3 [9]. Figure 5 shows the calculated density and enthalpy profiles for PPy with fluorine dopants. As observed, the density is clearly in the range of the expected value and the enthalpy ensures the system has been equilibrated.

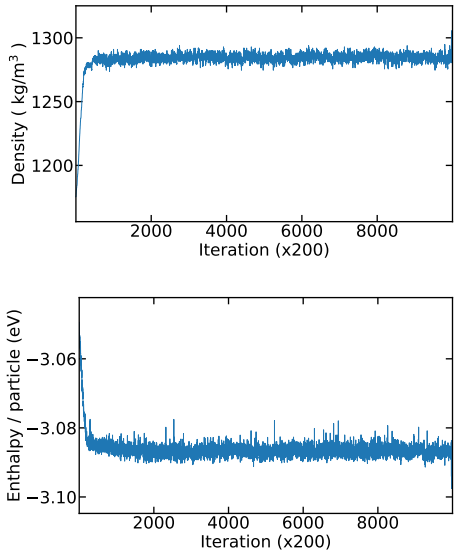

Figure 5: Density and enthalpy as a function of the MMC steps of a PPy system with fluorine dopants containing 1024 particles at 300 K and 1 atm.

4 CONCLUSION

We have performed a computational parallelization optimization of system containing a large number of PPy oligomers with atomic dopants by combining a distribution of loaded tasks between the CPU threads and a GPU installed in nodes of a computer cluster. The performance acceleration has been improved dramatically by creating a multidimensional table of values that can be interpolated to represent the interactions between particles instead of calculating these interactions from their analytical expressions. The performance gain is about 9.8 times the required execution time for the previous GPU implementation and 342 times better than the CPU only implementation. Our current computing approach is scalable with system size allowing for simulations of systems with more than 130,000 particles. Currently, this study is being extended to obtain the coarse grained model potential for PPy doped with halogen atoms other than chlorine.

REFERENCES

[1] T. K. Das and S. Prusty, "Review on conducting polymers and their applications", *Polym. Plast. Technol. Eng,* 51, 1487-1500, 2012.

[2] J. Pan, "Scaling up system size in materials simulation," *Nat. Comput. Sci.* 1, 95-95, 2021.

[3] C. Choudhury and P. Carbone, S. Roy, "Scalability of coarse-grained potentials generated from iterative Boltzmann inversion for polymers: Case study on polycarbonates," *Macromol. Theory Simul.* 25, 274-286, 2016.

[4] R. Wyrzykowski and B. K. Szymanski, "Taming next generation HPC systems: Run time system and algorithmic advancements," *Concurr. Comput.* 33, editorial, 2021. DOI:10.1002/cpe.6153

[5] A. Donaldson, G. Gopalakrishnan, N. Chong, J. Ketema, G. Li, et al., in *Advances in GPU Research and Practice, Emerging Trends in Computer Science and Applied Computing*, Ed. H. Sarbazi-Azad, chap. 1 "Formal analysis techniques for reliable GPU programming: current solutions and call to action," 2017, Morgan & Kaufmann, Boston.

[6] Y. Liang, X. Xing and Y. Li, "A GPU-based large-scale Monte Carlo simulation method for systems with long-range interactions," *J. Comput. Phys.* 338, 252-268, 2017.

[7] T.-S. Lee, D. S. Cerutti, D. Mermelstein, C. Lin, S. LeGrand, et al., "GPU-accelerated molecular dynamics and free energy methods in Amber18: Performance enhancements and new features," *J. Chem. Inf. Mod.* 58, 2043-2050, 2018.

[8] A. Yussuf, M. Al-Saleh, S. Al-Enezi and G. Abraham, "Synthesis and characterization of conductive polypyrrole: The influence of the oxidants and monomer on the electrical, thermal, and morphological properties," *Int. J. Polym. Sci.* 2018, 4191747, 2018. DOI:10.1155/2018/4191747.

[9] Y. Abere, G. Helmick and E. Blaisten-Barojas, "Modeling oxidised polypyrrole in the condensed phase with a novel force field," *J. Phys. Condens. Matter.* 34, 185701, 2022.

[10] X. Dong and Blaisten-Barojas, "Adaptive Tempering Monte Carlo Method," *J. Comput. Theor. Nano.* 3, 118-127, 2006.

[11] C. Hall, W. Ji, and E. Blaisten-Barojas, "The Metropolis Monte Carlo method with CUDA enabled graphic processing units," *J. Comp. Phys.* 258, 871–879, 2014.

[12] G. Helmick, Y. Abere and E. Blaisten-Barojas, "The caloric curve of polymers from the adaptive tempering Monte Carlo method," in *Advances in Parallel & Distributed Processing and Applications*: Eds. H. R. Arabnia, et al., 253-260, 2021, Springer, New York.