Molecular Dynamics Simulations of Biocorona Formation

Rongzhong Li, Cody A. Stevens and Samuel S. Cho

Abstract The development and advancement of nanomedicine has opened up many exciting, new applications of nanoparticles such as sensing, imaging, delivery, and therapy. However, their ability to readily enter cells and organelles that allow these nanomedical applications also opens up the possibility of unintended adverse nanotoxicity. The interaction between nanoparticles and biomolecules results in biocorona formation on the nanoparticle surface that is very different from adsorption of biomolecules on a flat surface. It remains a great challenge to understand the applications and risks associated with nanoparticles being in contact with biological systems beyond experimental methods that have limited resolution of the interactions and conformational changes involved. Recently, biomolecule-nanoparticle molecular dynamics (MD) simulations are becoming a viable approach for a detailed view of biocorona formation. In this review, we present the advantages and challenges of several MD simulation approaches for the study of biomolecule-nanoparticle interactions. In particular, we argue for the development of GPU-optimized MD simulations as a critical step in the study of biocorona formation. We discuss recent successes on how integrated computational and experimental studies are important to establish how the structure and functions of biomolecules are affected by nanoparticle interactions with the biomolecules.

1 Biomolecule-Nanoparticle Interactions

The development of nanomaterials and the characterizations of their properties have become a highly attractive subject that spans physics, chemistry, biology, and engineering [1–3]. Nanoparticle properties have opened up industrial products

R. Li · S.S. Cho (✉)
Department of Physics, Wake Forest University, Winston-Salem, NC 27106, USA
e-mail: choss@wfu.edu

C.A. Stevens · S.S. Cho
Department of Computer Science, Wake Forest University, Winston-Salem, NC 27106, USA

J. Suzuki et al. (eds.), *Modeling, Methodologies and Tools for Molecular and Nano-scale Communications*, Modeling and Optimization in Science and Technologies 9, DOI 10.1007/978-3-319-50688-3_10

involving nanomaterials, and there is particular interest in the development of biomedical applications involving nanoparticles due to their small size that allows them to interact with cellular machinery that was previously thought to be inaccessible [4]. On the other hand, as nanoparticles become widely used in many different industries, they also have the potential for unintended exposure to the environment and living matter that could result in deleterious toxicological effects. Their increased applications in daily life inevitably will lead to their accumulation in the environment [5], and their subsequent entry into biological systems, raising important questions about their nanotoxicity [6]. As such, there is a strong motivation to develop safe nanomedical applications that limit their negative effects. To understand the biological impact of nanoparticles, we must develop a molecular level understanding for their interactions with biological media.

Nanoparticles have very different properties from bulk materials. Due to their high surface area to volume ratio, nanoparticles have very active surface chemistry interactions with biological media that reduces their surface energy. As a result, their surfaces end up being rapidly covered by a complex layer of biomolecules [5]. The absorption of biomolecules to surfaces confers a new "biological identity" that results in different cellular responses. Of particular interest in recent studies is that formation of a "biological corona" on the nanoparticle surface that is very different from when biomolecules adsorb to a flat surface [7]. The biocorona formation is a dynamic process by which proteins and other biomolecules compete to interact with the nanoparticle surface. Recently, functionalized molecules conjugated on nanoparticles were shown to lose their function when placed in a biological medium presumably because protein corona formation screened the molecules from interacting with their intended target [8]. Therefore, the safe development of nanomedicine requires the study of nanoparticle interactions in the biological medium.

When biomolecules interact with nanoparticles, their structure, dynamics, and function are expected to change, which could further impact recognition of the biomolecules by their intended receptors. It is now well established that protein and RNA fold into specific structures in the cell to perform their biological functions [9, 10]. The main forces that stabilize the protein structure, such as the hydrophobic, electrostatic, van der Waals, and hydrogen bond interactions can also compete with the nanoparticle surface. The competition results in a perturbation of the protein structure such that it may not longer be able to perform its function.

For example, enzymes catalyze reaction in a well-defined, specific active site, but their structure may be destabilized such that the active site is significantly changed or completely lost upon interaction with the surface of a nanoparticle (Fig. 1a). Even if the active site were to be preserved, the substrate entrance or exit may be obstructed or destabilized. The end result is a loss of catalytic activity due to the presence of nanoparticles. The adsorption of lysozyme on a negatively charged silica nanoparticles resulted in 70% loss of α-helical structure, which accompanied 40% loss of catalytic activity [11].

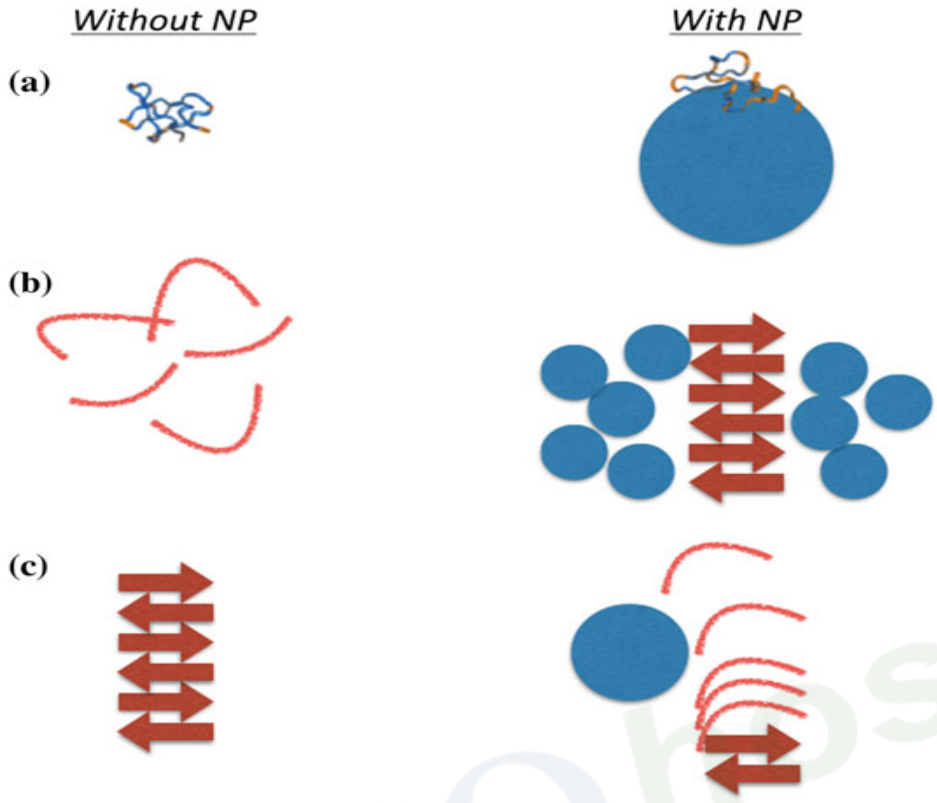

Fig. 1 Modes of biomolecular structural changes upon interaction with nanoparticles (*solid blue circles*). **a** A biomolecule can undergo structural changes upon binding to the nanoparticle surface, which can diminish or abolish their original biological function or introduce an unintended cellular response. **b** Biomolecular interaction with nanoparticles can induce fibril formation for peptides that would otherwise be unstructured. **c** Nanoparticle interaction can disrupt the formation of amyloid fibrils

A well known issue with protein folding is that some amyloidogenic proteins midfold and aggregate to form fibrils, which are thought to play important roles in neurodegenerative diseases such as Alzheimer's Parkinson's, and Huntington's diseases [12]. The introduction of nanoparticles may induce amyloidogenic peptides to form β-sheet structures [13] (Fig. 1b), which are the hallmark of fibril formation that lead to diseases.

The competition of native protein interactions with nanoparticles can also have a beneficial result too. Ikeda et al. showed that cholesterol-bearing pullulan nanogels inhibit protein aggregation in the case of Aβ peptides. The natively coiled Aβ peptides associated into protofibrils with a β-sheet structure that can act as nucleation centers for fibril formation. However, in the presence of hydrophilic nanogel particles, the β-sheet formation and therefore fibril nucleation was inhibited, presumably by competing with the hydrogen bonds that stabilize the β-sheet structure [14] (Fig. 1c).

2 Biomolecular MD Simulations Overview

Due to limitations of experimental instrument resolution, the molecular details of the protein-nanoparticle interactions and the formation of their biocorona remains poorly understood. A well-established approach for characterizing the molecular details of biological systems is molecular dynamics (MD) simulations, and they have recently been applied to protein-nanoparticle systems. In MD simulations, biomolecular systems are represented as sets of spherical beads that interact with one another and move in successive timesteps to result in a trajectory of its motion. Biomolecular dynamics, folding, and assembly mechanisms have been extensively studied using MD simulations (Fig. 2a). The physical description of a biomolecular system is determined by a potential energy function for the interactions.

The MD simulation algorithm then consists of two main portions: (1) computing the forces, based on the potential energy function, between interacting particles, and (2) updating of the position and velocities of the particles for the next timestep. The process is repeated over and over until the end of the MD simulation. In each timestep, a "snapshot" of the biomolecule's positions is obtained, and all of the

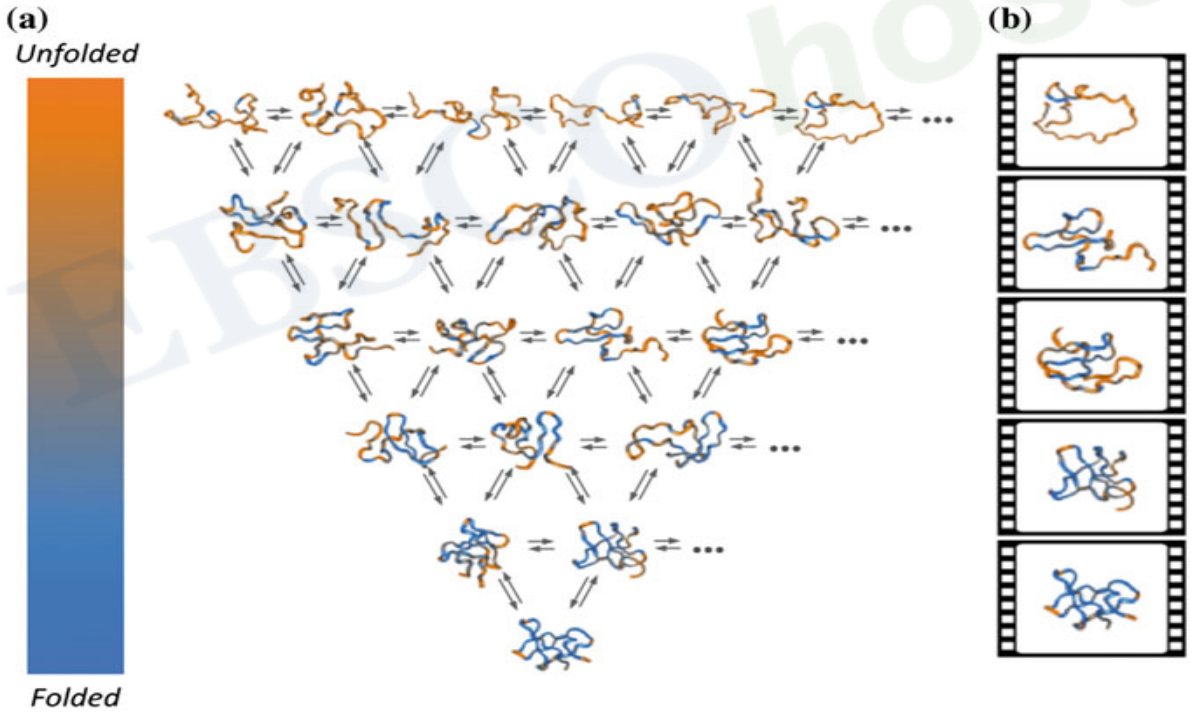

Fig. 2 Biomolecular folding mechanisms and MD simulations of their folding. **a** A schematic of a funneled energy landscape for folding where many possible unfolded states are directed towards a single, well-defined folded structure that corresponds to a function. For each protein structure from a coarse-grained Go-type MD simulation shown, the degree of local native structure formed is colored according to whether it is partially folded (*blue*) or unfolded (*orange*). **b** Representative structures from an MD simulation trajectory are shown

snapshots can be stitched together into a "movie" to result in a trajectory of the biomolecule's motion that is based on the potential energy function (Fig. 2b).

MD simulations are commonly implemented with an empirical force field. In this approach, a biomolecular system is described at atomistic resolution by an energy function that consists of bonds, angles, dihedral, electrostatic, and van der Waals interactions. The parameters for the energy function are derived from quantum mechanical calculations of model compounds or empirical data when reliable measurements are available. Popular atomistic empirical force field based MD simulation programs include CHARMM [15], AMBER [16], and NAMD [17], and the force field parameter sets are available for proteins [18], nucleic acids [19], and other biological systems.

In general, there are two main computational challenges for MD simulations of biomolecular systems to observe biologically relevant functional events (Fig. 3). The first is the system size because a large number of proteins interact with the nanoparticle. The second is the timescale required for the proteins to encounter the nanoparticle and interact with it. Large structural rearrangements or unfolding events are considered very challenging, even for relatively small systems (~100 residues) or small timescales (~μs-ms). Advanced sampling methods such as replica exchange can increase computational sampling by minimizing the time spent in trapped states, but the kinetic information is lost [20]. Others use advanced computing infrastructures such as the distributed computing Folding@Home approach [21] or the Anton supercomputer [22].

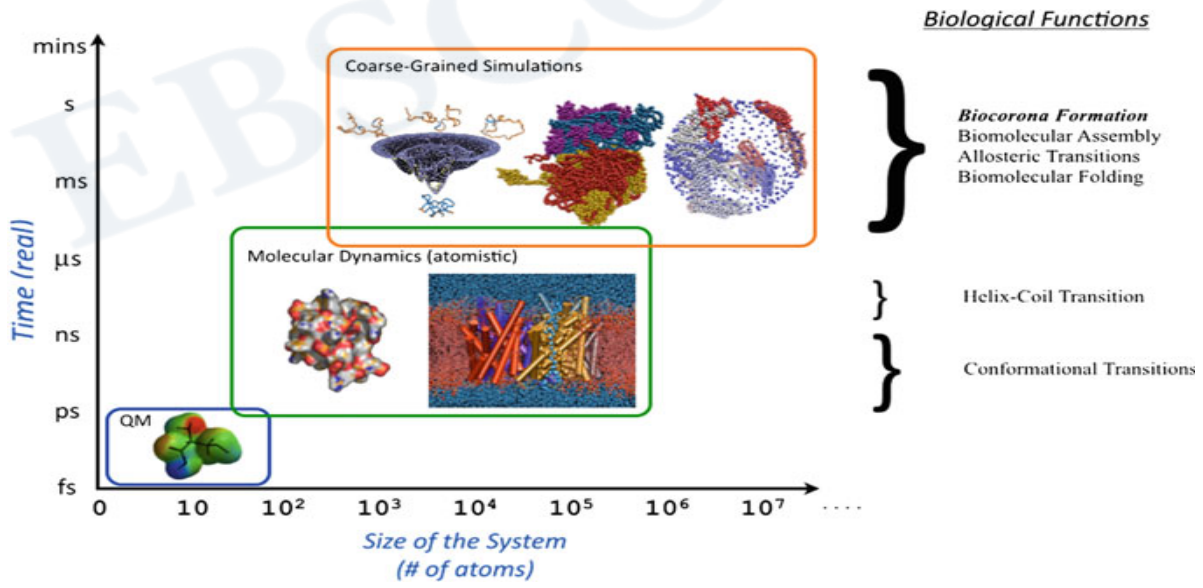

Fig. 3 Length- and time-scales of MD simulations and biological functions. **a** A graph of the approximate size of the system versus the length of time one can simulate using MD simulations using current computing standards using quantum mechanical, atomistic empirical force field, and coarse-grained Go-type MD simulations. **b** Some characteristic biological functions and the approximate corresponding timescales that can be observed in MD simulations

To increase the size and the timescales of the system one can simulate, some researchers use coarse-grained MD simulations where the residues or nucleic acids are represented as a single bead. A coarse-grained MD simulation approach that has been developed and rigorously tested is the Go-type model, which is defined by an energy function such that the long-range interactions are globally attracted to the native basin [23]. Based on the Funneled Energy Landscape Theory of biomolecular folding [10], the Go-type model MD simulation potential energy function consists of bonds, angles, dihedral, and Lennard-Jones terms whose global minimum corresponds to the folded state. For proteins, Go-type model MD simulations typically represent protein or RNA as a one bead per residue or nucleotide representation [24–26], but different degrees of coarse-graining such as a two bead per residue (backbone and sidechain) [27], three bead per nucleotide (base, sugar, and phosphate) [28–30], or even atomistic representations have been developed [31–33]. Although the size of the system and timescale one can simulate varies with the complexity of the Go-type model, researchers can readily perform MD simulations of 100–1,000s of residues, and the timescales are sufficiently large that one can observe multiple folding and unfolding events in a single trajectory.

Another simplification of MD simulations is to use discrete square-well potentials instead of a continuous potential for interactions between particles. In this discrete molecular dynamics (DMD) approach [34, 35], collision events determine the movement of the particles. A local search algorithm identifies particles that could possibly participate in a collision and only those particles are required to update their positions if a collision occurs. The fewer calculations required allows greater sampling at the expense of some accuracy.

To date, atomistic empirical force field MD simulations, coarse-grained Go-type model, and coarse-grained DMD simulations have been performed for protein-nanoparticle interactions.

3 MD Simulations of Protein-Nanoparticle Interactions

Silver nanoparticles (AgNPs) are widely produced commercially for antibacterial and antifungal applications and generating surface plasmon resonance for optical detecting and sensing [36, 37]. However, their cytotoxicity has been linked to their physical adsorption to cell membranes [38]. As such, the interactions of cytoskeletal proteins and other proteins in the cell with AgNP can significantly impact the cellular environment. Wen et al. performed dynamic light scattering, UV-Vis spectroscopy, and circular dichroism spectroscopy, hyperspectral imaging, and transmission electron microscopy on the cytoskeletal proteins actin and tubulin. In both cases, the cytoskeletal proteins readily interacted with citrate coated AgNPs, and they observed conformational changes of a decrease in α-helical content and an increase in β-sheet content upon binding to AgNPs. They suggested that the electrostatic, van der Waals, and hydrophobic interactions between the two species were the dominant interactions [39].

DMD simulations have been performed for a number of proteins interacting with a AgNP to characterize the consequence of biocorona formation. Ding et al. recently showed that for ubiquitin, interaction with AgNP results in a decrease in α-helical content and an increase in β-sheet too, and they observed a stretched exponential binding kinetics [40]. Interestingly, the secondary structure was only marginally affected in both experiments and simulation by interacting with AgNP for the firefly luciferase protein [41].

Auer et al. performed DMD simulations of peptides interacting with a nanoparticle [42]. They observe highly ordered β-sheet structure that eventually forms fibrils. Therefore, the presence of the nanoparticle catalyzes fibril formation by increasing the rate of peptide aggregate formation, and they attribute fibril formation enhancement to the increase in local concentration of the peptides upon interacting with the nanoparticle.

4 GPU-Optimized MD Simulations

Graphics Processing Units (GPUs) are becoming a widely accessible and mature computational architecture for performing MD simulations, and many studies have recently shown that MD simulations on GPUs can have a significant performance increase over the traditional CPU-only approach. Note there are other review articles on the development of GPU and can be found elsewhere [43–45], and we will briefly review the background here. Since the GPU architecture is significantly different from the CPU, implementing programs on the GPU often requires substantial changes in the implementation and the development of new algorithms that are optimized for the GPU architecture.

GPUs were originally designed for rendering computer graphics to offload and alleviate the computational burden of CPUs. The GPU is essentially an accelerator that is optimized for performing parallel floating-point calculations across individual geometric primitives due to the independent nature of rendering graphical images. As such, a standard GPU contains a few hundreds to thousands of independent cores that can work in parallel. This is in contrast to CPUs, which typically contain around 1–16 processors. The GPU processors are significantly slower than on a CPU, but there are so many of them so the overall floating point operation rate is higher. Therefore, the GPU architecture is ideally suited for parallel algorithms.

In order to utilize the resources found on the GPU to implemented general purpose parallel algorithms, NIVIDIA has developed their own programming language known as the Compute Unified Device Architecture (CUDA) so that software can be developed, compiled, and executed on NVIDIA GPUs. CUDA as a programming language is an extension of the C language with features that allow the program to execute code on both the CPU and GPU. The software instructions on the GPU architecture follow the Single Instruction Multiple Data (SIMD) paradigm where a kernel instruction performs the same instruction on different data (Fig. 4a). The GPU is able to execute code through the use of kernel calls that are

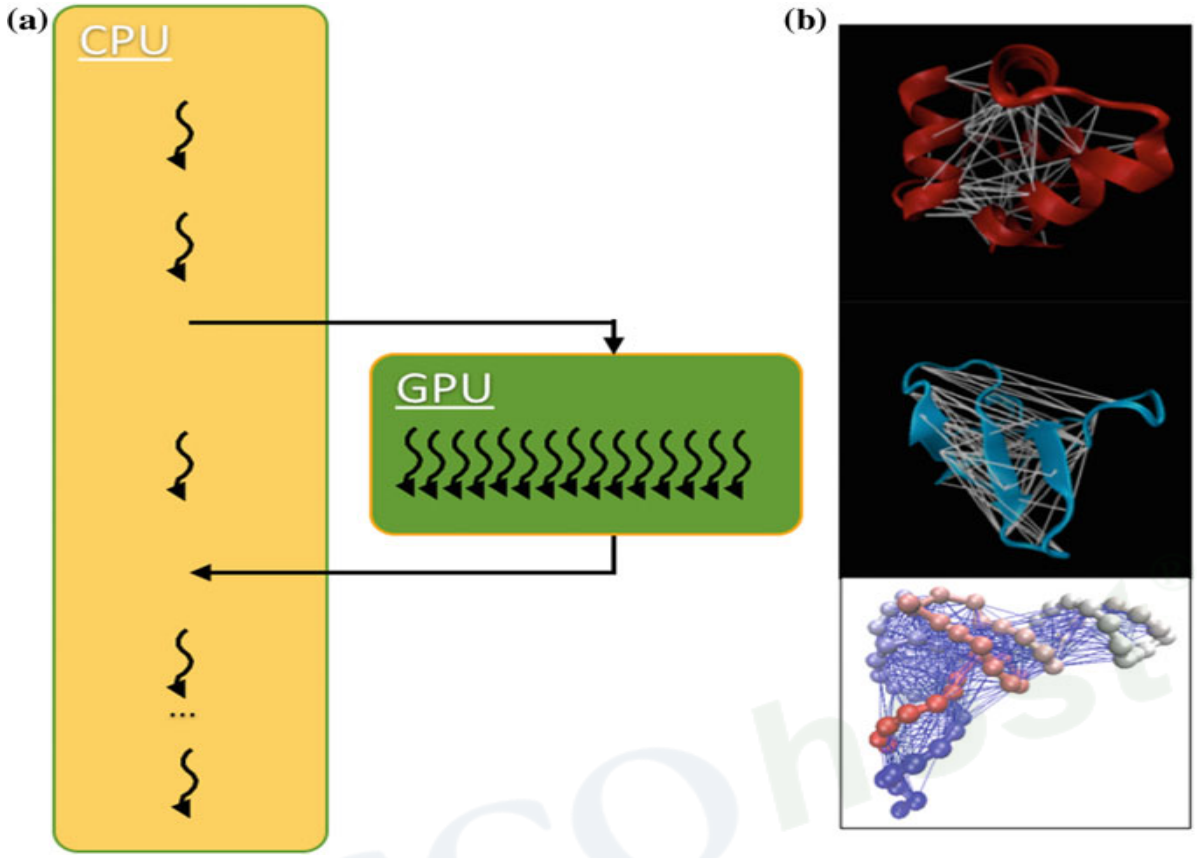

Fig. 4 Schematic of the GPU programming paradigm for MD simulations. **a** In a heterogeneous CPU-GPU architecture, a CUDA program executes on a CPU as a serial code. For portions of the code that is parallelizable, the program will transfer information to the GPU device and perform kernel operations (single instruction) in parallel on the (multiple) data using many independent threads. The data is then transferred back to the CPU. The MD simulation algorithm is parallelizable because the interactions in a single timestep are independent. **b** Representative α-helical (*top*) and β-sheet (*middle*) proteins and an RNA with their interactions shown with *lines*. These interactions are independent so the forces due to these interactions can be computed independently in a parallel algorithm

equivelant to functions for the CPU. When a kernel call is made on the CPU, the GPU allocates a certain number of threads that will each execute the kernel function in parallel. These threads are arranged in a grid system containing blocks of threads with each thread containing its own unique thread id. Threads are executed concurrently in warps where each thread executes all code within the kernel call. Before kernel calls are executed, all data that is used during parallel computation must be transferred to the GPU from the CPU. Similarly, all data that was generated by the GPU must be transferred back to the CPU once a kernel call is finished.

The MD simulation algorithm is well-suited for implementing on the GPU architecture. The calculation of the forces between interacting particles in a biomolecule is the most computationally demanding portion of the algorithm because

it considers all possible pairs of interacting beads, a $O(N^2)$ computation. The main advantage of performing MD simulations on the GPU is that the calculation of the forces between interacting particles is a parallelizable algorithm because those interactions are independent within a single timestep such that those calculations can be distributed among the different processors (p) of the GPU: $O(N^2)/p \sim O(N)$. Furthermore, cutoff methods that first evaluate whether an interaction is significant before actually computing the force, such as the well known Verlet neighbor list algorithm [46] or the cell list algorithm [47], can also improve performance without sacrificing accuracy if they can also be implemented on the GPU, as we will discuss below.

While there may be many benefits of performing simulations using CUDA on GPUs, these approaches also have their issues. The data transfer between the CPU and GPU use the PCI Express Bus, which is one of the main performance bottlenecks of any GPU program. Ideally, programs written for the GPU should minimize data transfer to avoid the performance cost. Another significant issue of GPU-optimized programs is the limited amount of memory available on the GPUs. The latest in GPUs offered by NVIDIA, the Titan X, is able to support 12 gigabytes of memory while most NVIDIA GPU models can only support 1-6 gigabytes of memory. If the problem size grows past the hardware memory limit, it is necessary to partition the data and (1) perform the calculation sequentially for every partition, (2) perform the calculation while other data is being transferred so that the GPU is performing calculations while the data transfer is occurring to hide the latency, or (3) use multiple GPU with OpenMP or MPI to perform the calculations, although a multi-GPU approach has its own issues because the information transfer across GPUs also is a performance bottleneck. There are now several atomistic empirical force field and coarse-grained Go-type or general MD simulation packages that are now available including ACEMD, AMBER, HOOMD-Blue, LAAMPS, NAMD, and SOP-GPU. Here, we will briefly review our own in-house coarse-grained Go-type (SOP model) MD simulation software for the GPU [48–50], but our discussion is general and applicable to all MD simulation codes for the GPU architecture (Fig. 5).

Software	Type	URL
ACEMD	Atomistic	http://multiscalelab.org/acemd
AMBER	Atomistic	http://ambermd.org
HOOMD-Blue	General	http://codeblue.umich.edu/hoomd-blue/
LAMMPS	General	http://lammps.sandia.gov
NAMD	Atomistic	http://www.ks.uiuc.edu/Research/gpu/
SOP-GPU	Coarse-grained	http://faculty.uml.edu/vbarsegov/gpu/sop/sop.html

Fig. 5 A table of MD simulation software available for the GPU architecture

We recently implemented our own version of a Go-type (SOP model) MD simulation software, and we introduced two novel GPU-optimized MD simulation algorithms, while inspired by CPU versions, is specifically optimized for the GPU architecture: (1) Parallel Verlet Neighbor List [48] and (2) Parallel Hybrid Neighbor/Cell List [50] Algorithms. In the original Verlet neighbor list algorithm, distance cutoffs were used to determine whether two particles were sufficiently close enough to have a significant interaction. Two lists were maintained, an outer "skin" layer that was updated every n timesteps and an inner "cutoff" layer such that the forces of only pairs of particles within the inner distance cutoff would be calculated. The distance cutoffs were chosen such that pairs of particles beyond the distance cutoff had forces that were essentially zero and no particle could leave the cutoff layer and move beyond the skin layer within the n timesteps.

When we compared our performance to a CPU version of the same MD simulation code, we observed an N-dependent speedup with about 30× speedup for the largest system we simulated (∼10,000 beads). For the smallest system we simulated, we actually observed a speed-down such that it took longer to perform the MD simulation on the GPU than on the CPU only. The performance gain from running the MD simulation on the GPU does not outweigh the cost of transferring the information to the GPU. We also developed a related Parallel Hybrid Neighbor/Cell List algorithm, and we observed about 10% speedup over the Parallel Verlet Neighbor List algorithm [50].

The Verlet neighbor list algorithm has been a staple of MD simulations for a very long time, but it is inherently serial because generating a subset list requires copying those pairs of interactions from the list of all pairs to an iteration dependent location in the subset list of pairs. We instead performed a parallel key-value sort of all pairs of interactions and placed all pairs of interactions to be copied to the subset list at the top. We then copied those pairs of interactions at the top in parallel over to the subset list. While the parallel version of the algorithm requires more steps, they can be performed entirely on the GPU in parallel.

Another significant issue of implementing software on the GPU involves accuracy of the floating point operations. Since the original purpose of the GPU is the rendering of graphical images, the floating point operations on older model GPUs did not need to be IEEE compliant like CPUs. In some implementations of MD simulations using early NVIDIA Tesla model GPUs, researchers observed an "energy drift" indicating that detailed balance was not being preserved due to errors in the calculations [51]. More recent models of GPUs are now IEEE compliant so this is no longer an issue. A related issue also exists where a large performance difference results between single and double precision calculations. MD simulations on CPUs have traditionally used double precision calculations because the performance difference was very minor. As such, there can be a performance gain if one chooses to implement MD simulations using single precision operations only.

We must note that the choice of double precision calculations over single precision calculations for MD simulations is arbitrary, and we are unaware of a significant reason for the choice. We therefore implemented our MD simulation code using single precision calculations only and double precision calculations only by

starting off with the same exact positions and velocities. Even though the initial conditions were identical, we do not expect the same results even if we ran two trajectories with double precision calculations because the order of the floating point operations would be different in two executions. We therefore used that situation as a control and compared differences to single versus double precision only MD simulations. To quantify our MD simulations, we computed order parameters that could be directly compared to experiments, specifically the radius of gyration that is measured in SAXS experiments and end-to-end distances that is measured in single molecule FRET experiments. We observed minimal differences between the single and double precision only MD simulations for our coarse-grained model [48]. Walker and coworkers introduced a mix single-double precision method for a GPU-optimized version of AMBER. They observed minimal energy drift and minimal differences with respect to RMSD and RMSF while preserving a performance that is similar to single precision only calculations [52].

5 GPU-Optimized MD Simulations of Biocorona Formation

We developed a novel GPU-optimized MD simulation approach for studying biocorona formation. Specifically, we simulated the interaction of a silver nanoparticle with apolipoprotein A-1, which is the main component of a high-density lipoprotein. In a CD spectroscopy experiment, with AgNP/apolipoprotein molar ratios of 1:300 and 1:600, the spectra reveals that the presence of AgNP reduces the ratio of α-helical content as compared to when there is no AgNP in the solution [53].

We developed a new computational model of nanoparticle interactions with proteins for the purpose of performing MD simulations of biocorona formation [53]. The model consists of 1 Ag nanoparticle (AgNP) and 15 apolipoproteins. We model the citrate-coated AgNP using 500 individual negatively charged spherical beads randomly distributed 10 nm from the center. And for the 243-residue apolipproteins, we developed a coarse-grained MD simulation based on well-established Go-type models of protein folding [23, 54].

In addition, we modeled the interactions between the AgNP and proteins by introducing two sets of terms: [1] excluded volume and [2] electrostatic interactions.

The excluded volume forbids the proteins getting too close and overlapping with the AgNP surface, and we modeled it using a hard sphere potential:

$$H_{\mathrm{EV}} = \sum_{i,j}^{\mathrm{protein}-\mathrm{NP}} \left(\frac{\sigma}{r_{\mathrm{ij}}}\right)^{12}$$

where r_{ij} is the distance between two interacting beads and σ is set to 3.8 Å. We also modeled the electrostatic interaction between the negatively charged AgNP and positive amino-acid residues with the Debye-Huckel potential:

$$H_{elec} = \sum_{i,j}^{protein-NP} \frac{z_i z_j e^2}{4\pi\epsilon_0\epsilon_r r} e^{-r/l_D}$$

where z_i and z_j are the charges of the interacting protein and nanoparticle beads and r is the distance between them. l_D is the Debye length, which can be tuned by the ion-concentration of 0.02 M [Na^+]. In our simulation, we used a relatively low ion-concentration for stronger attraction. We performed Langevin MD simulations in the underdamped limit (i.e., low friction coefficient) for effective sampling at 300 K (Fig. 6).

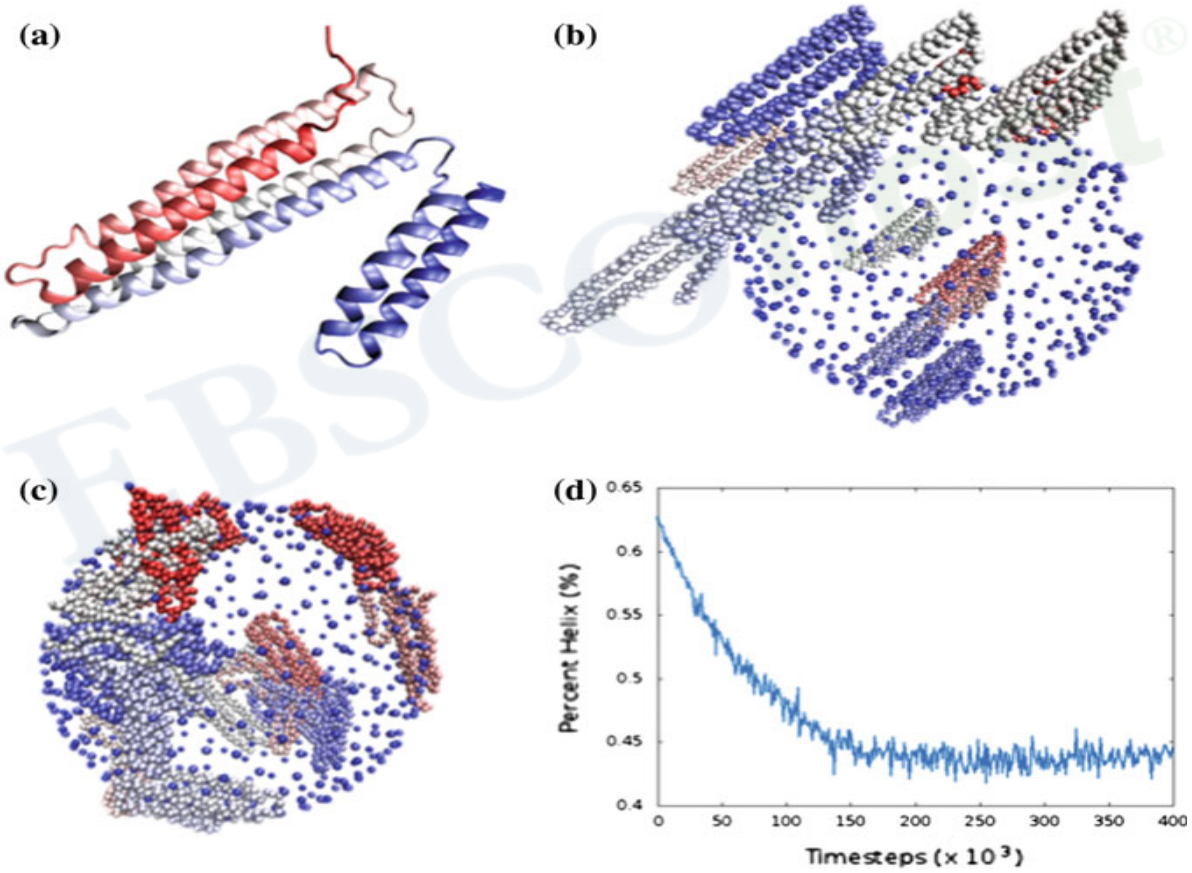

Fig. 6 MD simulations of biocorona formation on a GPU. **a** A ribbon diagram of apolipoprotein A-1. **b** The starting coordinates for coarse-grained MD simulations of 15 apolipoproteins around a negatively charged spherical model of AgNP composed of 500 beads. **c** A snapshot from GPU-optimized coarse-grained MD simulations of the AgNP-apolipoprotein biocorona. **d** A graph of the percent helicity of the apolipoprotein over a representative MD simulation trajectory. We observe a decrease in the helical content upon binding to the AgNP

The resulting system is comprised of 500 beads for the AgNP, and 3,645 beads (=243 residues/protein × 15 proteins) for the 15 apolipoproteins, which is computationally demanding even with a coarse-grained MD simulation approach. We therefore implemented our MD simulation model on a GPU-optimized parallel GPU code to improve the performance.

To quantify the secondary structure change during the biocorona formation, we monitored the torsional angles formed among the four successive beads in helical region, and we define an α-helical angle to be between 45°–60°. In our MD simulations, we observe a decrease in the percentage of the α-alpha helical angles from 65 to 45%, which is similar to experimentally observed values from CD spectra [53].

6 Conclusions

The development of nanomedicine and the widespread use of nanoparticles for industrial use have opened up new advances but it is also pressing to evaluate their biological exposure to determine how unintentional accumulation and exposure to nanoparticles could lead to toxicological effects due to their interaction with biological systems. Experiments have shown that nanoparticles readily interact with the biological medium to form a biocorona that assumes a new biological identity. To understand the consequences, we must evaluate nanoparticle interactions with biomolecules using an interdisciplinary approach that spans physics, chemistry, biology, engineering, and computer science.

The relatively large size of the biocorona requires new and advanced approaches for the study of their formation from MD simulations. Coarse-grained MD simulations are powerful tools that can access the timescales necessary to study these events while still reproducing experimental observables. More detailed description of the phenomenon requires advanced computing power, and the GPU architecture has already demonstrated to be a valuable tool for the study of biocorona formation that can reproduce experimental observables and provide new insight at a molecular resolution. These computational advances can give much needed biophysical insight into our understanding of biological and environment consequences of nanomaterial interactions.

Acknowledgements The National Science Foundation (CBET-1232724) financially supported this work. RL and SSC acknowledge financial support from the Wake Forest University Center for Molecular Communication and Signaling (CMCS). CAS was supported by a CMCS graduate research fellowship. SSC appreciates fruitful conversations with Pu-Chun Ke.

References

1. Alivisatos AP (1996) Perspectives on the physical chemistry of semiconductor nanocrystals. J Phys Chem 100:13226–13239
2. Dujardin E, Mann S (2002) Bio-inspired materials chemistry. Adv Mater 14:775–788
3. Nirmal M, Brus L (1999) Luminescence photophysics in semiconductor nanocrystals. Acc Chem Res 32:407–414
4. Ye D et al (2013) Nanoparticle accumulation and transcytosis in brain endothelial cell layers. Nanoscale 5:11153–11165
5. Cedervall T et al (2007) Understanding the nanoparticle–protein corona using methods to quantify exchange rates and affinities of proteins for nanoparticles. Proc Natl Acad Sci 104:2050–2055
6. Schrurs F, Lison D (2012) Focusing the research efforts. Nat Nanotechnol 7:546–548
7. Morriss-Andrews A, Bellesia G, Shea J-E (2011) Effects of surface interactions on peptide aggregate morphology. J Chem Phys 135:085102–085109
8. Salvati A et al (2013) Transferrin-functionalized nanoparticles lose their targeting capabilities when a biomolecule corona adsorbs on the surface. Nat Nanotechnol 8:137–143
9. Anfinsen CB (1973) Principles that govern the folding of protein chains. Science 181:223–230
10. Wolynes PG, Onuchic JN, Thirumalai D (1995) Navigating the folding routes. Science 267:1619–1620
11. Vertegel AA, Siegel RW, Dordick JS (2004) Silica nanoparticle size influences the structure and enzymatic activity of adsorbed lysozyme. Langmuir 20:6800–6807
12. Dobson CM (1999) Protein misfolding, evolution and disease. Trends Biochem Sci 24: 329–332
13. Linse S et al (2007) Nucleation of protein fibrillation by nanoparticles. Proc Natl Acad Sci 104:8691–8696
14. Ikeda K, Okada T, Sawada S, Akiyoshi K, Matsuzaki K (2006) Inhibition of the formation of amyloid β-protein fibrils using biocompatible nanogels as artificial chaperones. FEBS Lett 580:6587–6595
15. MacKerell AD et al (1998) All-atom empirical potential for molecular modeling and dynamics studies of proteins. J Phys Chem B 102:3586–3616
16. Weiner PK, Kollman PA (1981) AMBER: assisted model building with energy refinement. A general program for modeling molecules and their interactions. J Comput Chem 2:287–303
17. Phillips JC et al (2005) Scalable molecular dynamics with NAMD. J Comput Chem 26: 1781–1802
18. MacKerell AD, Feig M, Brooks CL (2004) Improved treatment of the protein backbone in empirical force fields. J Am Chem Soc 126:698–699
19. Denning EJ, Priyakumar UD, Nilsson L, Mackerell AD (2011) Impact of 2'-hydroxyl sampling on the conformational properties of RNA: update of the CHARMM all-atom additive force field for RNA. J Comput Chem 32:1929–1943
20. Sugita Y, Okamoto Y (1999) Replica-exchange molecular dynamics method for protein folding. Chem Phys Lett 314:141–151
21. Voelz VA, Bowman GR, Beauchamp K, Pande VS (2010) Molecular simulation of ab initio protein folding for a millisecond folder NTL9(1-39). J Am Chem Soc 132:1526–1528
22. Dror RO, Dirks RM, Grossman JP, Xu H, Shaw DE (2012) Biomolecular simulation: a computational microscope for molecular biology. Annu Rev Biophys 41:429–452
23. Clementi C, Nymeyer H, Onuchic JN (2000) Topological and energetic factors: what determines the structural details of the transition state ensemble and "en-route" intermediates for protein folding? An investigation for small globular proteins. J Mol Biol 298:937–953
24. Chavez LL, Onuchic JN, Clementi C (2004) Quantifying the roughness on the free energy landscape: entropic bottlenecks and protein folding rates. J Am Chem Soc 126:8426–8432

25. Karanicolas J, Brooks CL (2002) The origins of asymmetry in the folding transition states of protein L and protein G. Protein Sci 11:2351–2361
26. Hyeon C, Dima RI, Thirumalai D (2006) Pathways and kinetic barriers in mechanical unfolding and refolding of RNA and proteins. Structure 14:1633–1645
27. Cheung MS, Garcia AE, Onuchic JN (2002) Protein folding mediated by solvation: Water expulsion and formation of the hydrophobic core occur after the structural collapse. Proc Natl Acad Sci USA 99:685–690
28. Hyeon C, Thirumalai D (2005) Mechanical unfolding of RNA hairpins. Proc Natl Acad Sci 102:6789–6794
29. Cho SS, Pincus DL, Thirumalai D (2009) Assembly mechanisms of RNA pseudoknots are determined by the stabilities of constituent secondary structures. PNAS 106:17349–17354
30. Li R, Ge HW, Cho SS (2013) Sequence-dependent base-stacking stabilities guide tRNA folding energy landscapes. J Phys Chem B 117:12943–12952
31. Li L, Shakhnovich EI (2001) Constructing, verifying, and dissecting the folding transition state of chymotrypsin inhibitor 2 with all-atom simulations. PNAS 98:13014–13018
32. Whitford PC et al (2009) An all-atom structure-based potential for proteins: bridging minimal models with all-atom empirical forcefields. Proteins Struct Funct Bioinforma 75:430–441
33. Feng J, Walter NG, Brooks CL (2011) Cooperative and directional folding of the preQ 1 riboswitch aptamer domain. J Am Chem Soc 133:4196–4199
34. Dokholyan NV, Buldyrev SV, Stanley HE, Shakhnovich EI (1998) Discrete molecular dynamics studies of the folding of a protein-like model. Fold Des 3:577–587
35. Proctor EA, Ding F, Dokholyan NV (2011) Discrete molecular dynamics. Wiley Interdiscip Rev Comput Mol Sci 1:80–92
36. Choi O, Hu Z (2008) Size dependent and reactive oxygen species related nanosilver toxicity to nitrifying bacteria. Environ Sci Technol 42:4583–4588
37. Jin X et al (2010) High-throughput screening of silver nanoparticle stability and bacterial inactivation in aquatic media: influence of specific ions. Environ Sci Technol 44:7321–7328
38. Zhang W, Yao Y, Sullivan N, Chen Y (2011) Modeling the primary size effects of citrate-coated silver nanoparticles on their ion release kinetics. Environ Sci Technol 45: 4422–4428
39. Wen Y et al (2013) Binding of cytoskeletal proteins with silver nanoparticles. RSC Adv 3:22002–22007
40. Ding F et al (2013) Direct observation of a single nanoparticle–ubiquitin corona formation. Nanoscale 5:9162–9169
41. Käkinen A et al (2013) Interaction of firefly luciferase and silver nanoparticles and its impact on enzyme activity. Nanotechnology 24:345101
42. Auer S, Trovato A, Vendruscolo M (2009) A condensation-ordering mechanism in nanoparticle-catalyzed peptide aggregation. PLoS Comput Biol 5:e1000458
43. Anderson JA, Lorenz CD, Travesset A (2008) General purpose molecular dynamics simulations fully implemented on graphics processing units. J Comput Phys 227:5342–5359
44. Liu W, Schmidt B, Voss G, Müller-Wittig W (2008) Accelerating molecular dynamics simulations using graphics processing units with CUDA. Comput Phys Commun 179: 634–641
45. Xu D, Williamson MJ, Walker RC (2010) In: Wheeler RA (ed) Annual reports in computational chemistry, Elsevier, pp 2–19
46. Verlet L (1967) Computer "experiments" on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules. Phys Rev 159:98
47. Allen MP, Tildesley DJ (1989) Computer simulation of liquids. Oxford University Press, USA
48. Lipscomb TJ, Zou A, Cho SS (2012) Proceedings of the ACM conference on bioinformatics, computational biology and biomedicine, BCB'12. ACM, New York, NY, USA, pp 321–328
49. Proctor AJ, Lipscomb TJ, Zou A, Anderson JA, Cho SS (2012) 2012 ASE/IEEE international conference on biomedical computing (BioMedCom), pp 14–19

50. Proctor AJ, Stevens CA, Cho SS (2013) Proceedings of the international conference on bioinformatics, computational biology and biomedical informatics, BCB'13. ACM, New York, NY, USA, pp 633:633–633:640
51. Bauer BA, Davis JE, Taufer M, Patel S (2011) Molecular dynamics simulations of aqueous ions at the liquid–vapor interface accelerated using graphics processors. J Comput Chem 32:375–385
52. Le Grand S, Götz AW, Walker RC (2013) SPFP: speed without compromise—a mixed precision model for GPU accelerated molecular dynamics simulations. Comput Phys Commun 184:374–380
53. Li R et al (2013) Computational and experimental characterizations of silver nanoparticle-apolipoprotein biocorona. J Phys Chem B 117:13451–13456
54. Cho SS, Levy Y, Wolynes PG (2006) P versus Q: Structural reaction coordinates capture protein folding on smooth landscapes. Proc Natl Acad Sci 103:586–591