pyPolyBuilder: Automated preparation of molecular topologies and initial configurations for molecular dynamics simulations of arbitrary supramolecules.

Mayk C. Ramos,[†,‖] Patrick K. Quoika,[‡,‖] Vitor A. C. Horta,[¶] Douglas M. Dias,[§] Elan G. Costa,[§] Jorge L. M. do Amaral,[§] Luigi M. Ribeiro,[§] Klaus R. Liedl,[‡] and Bruno A. C. Horta[*,†]

†*Instituto de Química, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 21941-909, Brazil*

‡*Institute for General, Inorganic and Theoretical Chemistry, Center for Molecular Biosciences Innsbruck (CMBI), University of Innsbruck, A-6020 Innsbruck, Austria*

¶*Insight Centre for Data Analytics at Dublin City University, Dublin, Ireland*

§*Dept. of Electronics and Telecomm. Eng., State University of Rio de Janeiro, Rio de Janeiro 20550-900, Brazil*

‖*Contributed equally to this work*

E-mail: bruno@iq.ufrj.br

Abstract

The construction of a molecular topology file is a prerequisite for any classical molecular dynamics simulation. However, the generation of such a file may be very challenging at times, especially for large supramolecules. While many tools are available

to provide topologies for large proteins and other biomolecules, the scientific community researching non-biological systems is not equally well equipped. Here, we present a practical tool to generate topologies for arbitrary supramolecules: The pyPolyBuilder. In addition to linear polymer chains, it also provides the possibility to generate topologies of arbitrary, large, branched molecules, such as *e.g.* dendrimers. Furthermore, it also generates reasonable starting structures for simulations of these molecules. pyPolyBuilder is a standalone command-line tool implemented in python. Therefore, it may easily be incorporated in persisting simulation pipelines on any operating systems and with different simulation engines.

1 Introduction

General motivation Nowadays, molecular systems are being synthesized in order to fulfill the technological demand found in several areas of nanotechnology.[1–3] The development of specific and complex systems poses a challenge to the following up of analytical techniques to study these novel molecules.[4,5] Designing nanostructures with handcrafted properties following a naive trial and error approach does not promise to be of huge success. This leads to an increasing demand for computational methods to investigate such complex systems. Nevertheless, treating arbitrary complicated supramolecules leads to a number of technical issues for computational scientists, as elaborated on below.

MD simulations In the course of the last decades, molecular dynamics (MD) simulation has been developed to be the state-of-the-art technique for studying the dynamics of large molecular systems.[6–8] With the continuous improvement of the available computational power, the scales of time and size of systems which may possibly be investigated computationally is ever-growing. Resolving physical processes of systems in nanometer size and nanoseconds time scale have long been available and as of now science is approaching even higher magnitudes.[9] Given this development, large supramolecular systems are becom-

ing accessible computationally.[8] MD Simulations may provide insights about the molecular behavior of such complex systems and support the understanding of experimental data.

In order to carry out an MD simulation, one must provide the initial coordinates and the molecular topology of the system of interest. The topology consists of lists defining the bonded and non-bonded interactions according to a chosen force field.

In most (MD) simulation packages such as GROMOS,[10,11] GROMACS,[12] AMBER,[13] the parameters are not directly read from the parameter file, but are rather from a system specific file, namely molecular topology file (MTF). An MTF typically includes information about a given molecular system such as the atom types, partial charges, masses, bond connectivity and the associated parameters (taken from the parameter file). The number of entries in the MTF can be huge and preparing such a file by hand may be cumbersome. Thus, in order to reduce the human effort and to avoid input errors, these MTFs are constructed automatically. Usually, simulation engines provide additional software for this preparation. For instance, in the GROMOS software,[10,11] the `make_top` program generates the MTF based on a parameter file and a building block file. Similarly, GROMACS[12] makes use of tools such as `pdb2gmx` to perform such a task. This MTF, together with a coordinate file and an input instructive file are the three basic input files for carrying out a simulation. While the MTF may be technically human-readable for simple systems, for large systems - containing a large number of atoms, or rather complex molecular structures - these topologies may become arbitrarily extensive and complicated. Without specific tools, providing a topology may be quite challenging or even literally impossible to handle by hand.

Supramolecules Among these, dendrimers and highly branched polymers (HBPs) are of special interest. Dendrimers[14] are separable into distinct structural subunits or *building blocks* (BBs). Generally, three types of these BBs compose a dendrimer: The **core central block** - which may be understood as the origin of the growth of the structure. From there, **intermediary blocks** are attached. This may happen in multiple shells. Lastly, the

terminal blocks[15] (see Figure 1). For each new shell of intermediary BBs, we speak of an additional *generation* of the dendrimer. Commonly, the size of dendrimers is described by the *generation number*. The three dimensional structures of these dendrimers lead to exceptional properties.[16,17] In contrast, HBPs are not as strictly defined as dendrimers. These molecules are polymers of highly branched, mostly globular structure. Their structural topology may be described in multiple different ways. Often, a set of structural indices is given for this purpose (such as the degree of polymerization, or the degree of branching). Unfortunately, these descriptions are not always fully distinct. These systems are intrinsically random and polydisperse. As a result, they may exhibit various specific properties important in different applications.[18,19] Such complex supramolecules - dendrimers, in particular - have been of interest for cancer diagnoses and for the development of delivery systems for drugs and genes.[20–22]

Literature & available alternatives To minimize the human bias in the process, the preparation of supramolecules for simulations may be automatized. This includes the construction of an MTF and the generation of initial coordinates. Therefore, various different approaches have been implemented in the past. In most cases, available tools dealt with only one of these two issues. Here we give an extensive list of software for this purpose (with no claim of completeness): PRODRG package,[23,24] Sequential addition method,[25] Self-avoiding random walker method,[26] Antechamber,[27] GBstudio,[28] Protein/membrane complex,[29] MKTOP,[30] ATB,[31–33] Cellulose builder,[34] LAMBADA/InflateGRO2,[35] among others. Each of the aforementioned programs is specialized and focuses on a specific category of systems. Furthermore, various implementations of automated procedures to specifically generate topologies of dendrimers and HBPs have been published. For the interested reader, we give an overview of the functionality of a selection of these modules in the SI.

In summary, two main approaches for automatically constructing MTFs can be found in the literature. One approach considers the combination of pre-built molecular building

blocks. Usually, such tools are available within the MD software distribution for the most commonly studied biomolecules, *e.g.* aminoacids or nucleic acids. In these cases, the rules for connecting these building blocks are well defined. Therefore, specific classes of macromolecules such as proteins, nucleic acids and carbohydrates may be constructed conveniently. The alternative approach considers the creation of MTFs on the basis of molecular three dimensional structures (*e.g.* PDB files) and is usually applied for small molecules. This is specially important in drug-design applications. Examples of such structure-based topology builders are: PRODRG,[23,24] ANTECHAMBER,[27] MKTOP[30] and ATB.[31–33] Although, this method has also been applied for macromolecules,[36] the building-block approach is usually preferred in this case.

Outlook paper Here, we present **pyPolyBuilder**: a python application that performs the preparation of MTF and initial coordinates for MD simulations of macromolecules (*e.g.* linear, branched and hyperbranched polymers, such as dendrimers).

Generally, pyPolyBuilder constructs a large topology from a combination of BBs according to a connectivity file. Thereby, it enables the user to realize arbitrary networks of building blocks. In the words of graph theory, it may be described as a builder of network topologies, in which the nodes are molecular building blocks. In other words, it builds a *topology of topologies*. pyPolyBuilder may be used to construct linear polymers, but also arbitrarily branched polymers. While the former may be visualized as a simple linear network, the latter may be structured like *e.g.* a *tree*, or *bus* network. The preparation of symmetric hyperbranched polymers such as dendrimers is organized in a specific module, *i.e.*, the *dendrimer module* of pyPolyBuilder. It may be regarded as a specification of the general procedure — which we call *network module* —, since a dendrimer may be described as a star-shaped network. In the following, we will explain the general philosophy, functionality and implementation. Furthermore, we will show a number of illustrating examples for different use cases.

2 Development and Features

The development of pyPolyBuilder was motivated by our need to construct molecular topologies for highly complex, yet structurally subdividable systems, such as heterodendrimers and dendrimer polymers (dendrimers, which are connected to each other), but also linear polymer chains of variable length. Below, we outline the architecture and functionality of pyPolyBuilder.

2.1 Philosophy - Linkage of Building Blocks

The program is designed to construct virtually any kind of macromolecular topology, which can be assembled from simplified topologies of *small* molecular fragments, which can be easily generated. The problem becomes challenging when many of such fragments are desired to be linked. pyPolyBuilder takes care of the linkage. To keep the handling of pyPolyBuilder as simple as possible, we decided to stick to established formats of topology files and to a very regularized API. Depending on the nature of the molecular system to be assembled, the program may be run in either the *dendrimer* or the *network mode*. For complex systems, the user may also combine these modes. Below, we outline the general functionality of the program. Subsequently, we explain the two different modes.

2.2 Topology Building

pyPolyBuilder uses an object-oriented framework: Each atom, bond, angle and dihedral angle is a distinct object, which is furthermore linked to other objects of corresponding classes. For instance, angles are constructed over two consecutive bonds, and torsional dihedral angles over three consecutive bonds. The parameters that are considered to belong to the building block (intra-block parameters) can be read either from the BB definition or guessed by pyPolyBuilder. However, the parameters of the newly generated degrees of freedom (inter-block parameters) are assigned by pyPolyBuilder exclusively based on a parameter list that must be provided by the user. Wherever necessary, improper dihedral angles are introduced. Therefore, neighbors of neighboring atoms of a certain reference atom are evaluated.

Depending on the nature of the supramolecule to be built — linear polymers, asymmetrically branched polymers, or dendrimers — pyPolyBuilder follows slightly different approaches for the growth of the topology, see Figure 1. These will be outlined in the following sections.

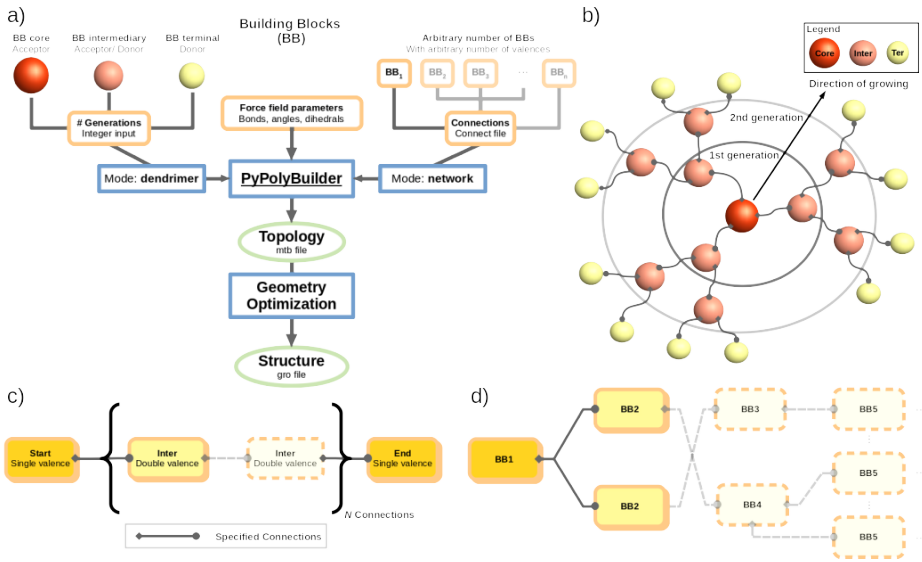

Figure 1: Flowchart of pyPolyBuilder. (a) general operating principle. Inputs (BBs, FF parameters and connectivity file) are shown in orange, outputs (topology and structure) in green, and functional units of pyPolyBuilder in blue. The inputs specific for the dendrimer and network modules are shown in the top-left and top-right positions, respectively. Subfigures b-d depict the operating modes of pyPolyBuilder. (b) outlines the construction of dendrimers with the pyPolyBuilder, showing the core (red), intermediary (orange) and terminal BBs (yellow). (c) and (d) illustrate the network module for linear and more complex (cyclic/branched) chains, respectively.

Network Module

The network module of pyPolyBuilder enables the user to connect an arbitrary number of BBs in any manner. Therefore, it can be used to construct supramolecules of arbitrary complexity and degree of branching. Hence, this module is an extremely versatile machinery to construct such systems under the concept of *topology of topologies*. We show example schemes of different complexity on how to use pyPolyBuilder with the network module in Figure 1.

Conveniently, linear polymer chains may be constructed from a consecutive series of BBs. Therefore, the terminal blocks need to have a single valence, whereas the intermediate blocks need to have two valences to be connected. For such systems, a starting and an ending block, as well as the repeating unit must be specified. This scheme is visualized in Figure 1c. The versatility of the network module does not end there. Given the specific connections, any number of different BBs may be connected as desired. We show an example of a construction of a more complex polymer in Figure 1d.

The BBs are connected at specified atoms. Therefore, under the *network* mode, the user needs to list the connections in a separate input file. There, the identifier of the accepting and the donating BB and the respective atoms are given for each connection. Thus, pyPolyBuilder processes this list in sequence.

Dendrimer Module

Different from linear polymers, dendrimers grow outwards from a center block. Although it would be possible to construct a dendrimer using the general purpose *network* mode, it would need a very complex connectivity file. In order to facilitate this process, pyPolyBuilder offers the possibility of exploring the dendrimer symmetry, resulting in a simpler input.

In the *dendrimer* mode, instead of specifying a connectivity file, the branching points are defined within the building blocks. For instance, a tetravalence core block must have a *branch* block the specification of four open valencies. Similarly, the intermediary and terminal blocks must also specify its branching points. Hence, instead of providing a connectivity file, the user provides the generation number, which defines how many layers of intermediate blocks are present. A schematic representation is depicted in Figure 1b. Conventionally, in the *branch* definition of each BB, connections between blocks consist of either a *donor* and an *acceptor* atom. This nomenclature enables the specification of dedicated branching points for every block and also a direction of growth. While the core block only *accepts* connections, the terminal blocks only *donate* connections, whereas intermediary blocks have dedicated donor and acceptor atoms.

2.3 Geometry Optimization

In order provide a reasonable starting structure for an MD simulation, we use the constructed topology to generate an optimized geometry. In order to avoid entangled structures, the geometry construction is carried out in a two-step process. First, the torsional degrees of freedom are optimized to maximize the sphericity. This is carried out by constructing a Z-matrix based on the topology and optimizing only the torsional degrees of freedom therein using a genetic algorithm.[37–40] This procedure is explained in the SI. As a result, a spread conformation of the constructed supramolecule is obtained.

The second step of the optimization relies on a steepest decent algorithm to minimize the potential energy with respect to all degrees of freedom and considering bonded and non-

bonded interactions. These generated geometry serve as appropriate starting point for MD simulations.

3 Application Examples

To demonstrate the versatility of pyPolyBuilder, we provide illustrative examples below, starting from linear polymer chains and concluding with highly complex branched polymeric systems. These examples are also included as tutorials in the public github repository: `https://github.com/mssm-labmmol/pypolybuilder`

3.1 Linear Polymers

Poly(*N*-Isopropylacrylamide) - PNIPAAM

pNIPAAM is a polymer known to exhibit a liquid-gel phase transition with LCST.[41] It has been considered in many computational studies over the last years[42–46] We constructed pNIPAAM polymers of different sizes (5-, 16- and 64-mer) using the 2016H66 force field.[47] The generated configurations are depicted in Figure 2.

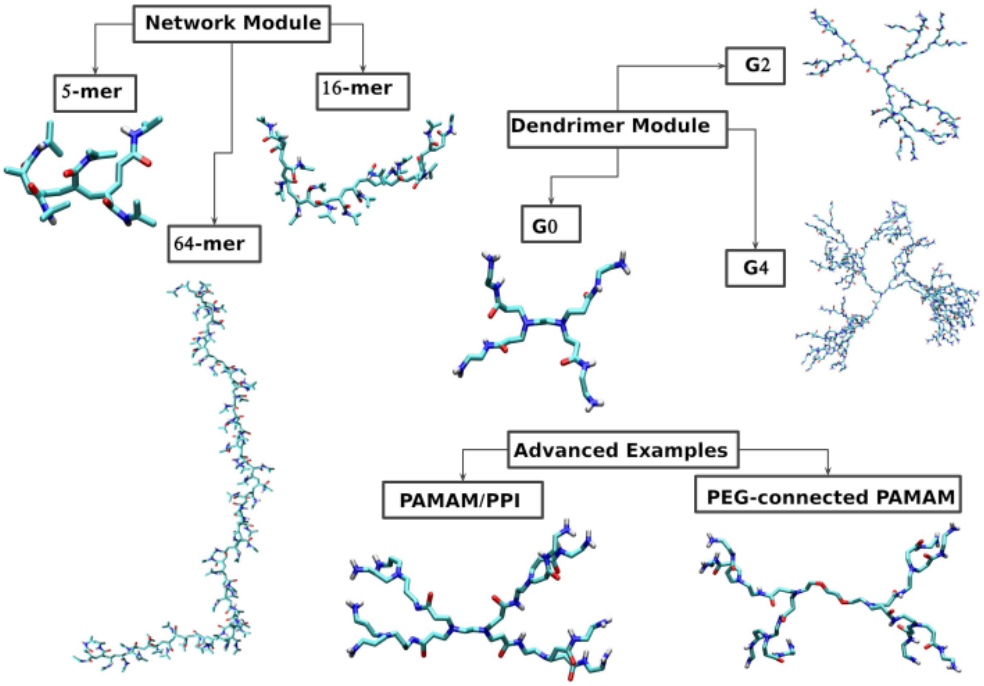

Figure 2: Illustrative examples of molecular systems generated using pyPolyBuilder. The left corner shows N-isopropylacrylamide polymers (pNIPAAM) of different lengths. The examples of the top right illustrate PAMAM dendrimers of increasing generation numbers. The bottom-right corner shows a PAMAM/PPI Janus dendrimer and a PEG-connected PAMAM dendrimer.

3.2 Branched Polymers

Below we illustrate the construction of branched polymers, such as *e.g.* dendrimers. We distinguish between homodendrimer for a pure dendrimer (only one type of core, intermediate and terminal blocks) and heterodendrimers (mixing of at least one of the dendrimer components). In addition, we show a more advanced example of a hyperbranched polymer.

Homodendrimer: Poly(amidoamine) - PAMAM

PAMAM dendrimers have BBs containing amide and amine groups. They represent an important class of dendrimers[4,15,48] The example shows a first generation (G1) PAMAM dendrimer (Figure 2). See Ref.[15] for more details.

Heterodendrimer: Poly(amidoamine) and Poly(propyleneimine) - PAMAM-PPI

In this example, the dendrimer contains BBs from both PAMAM and poly(propyleneimine) (PPI) dendrimers (Figure 2). PAMAM/PPI dendrimers have been of increasing interest in the development of nanomaterials.[15,49] In particular, they may be used in cancer diagnosis and therapy.[21] Such heterodendrimer is usually referred to as a *Janus-type* construct.[50,51] In order to achieve this construct, we grew the PAMAM half first, leaving 2 open valencies in the PAMAM core, which were subsequently used to grow the PPI half.

Hyperbranched Polymer: Poly(amidoamine) and Poly(ethyleneglycols) - PAMAM-PEG-separated Dendrimer

Branched heteropolymers of PAMAM and PEG have been used for the development of nonviral carriers for drug or gene delivery. In particular, PEG-functionalized PAMAM dendrimers have been studied.[52,53] In this example, a non-standard PAMAM-PEG heterodendrimer is shown, having two halves of PAMAM interconnected by a PEG linker (Figure 2).

4 Discussion and Conclusion

pyPolyBuilder is a versatile and flexible tool for the preparation of arbitrary supramolecular structures. It is able to generate molecular topologies and plausible starting structures for MD simulations. Important aspects of pyPolyBuilder with respect to other available tools are: (*i*) it follows a standalone approach; (*ii*) it can be extended to include compatibility with any MD engine and force field; (*iii*) it works as a command-line tool and is easily integrated in automated software pipelines. Moreover, pyPolyBuilder is modular, fast and easy to use. Due to the development in Python, it is portable across different computer architectures. Furthermore, since Python is a widely used language and state-of-the-art for pre- and postprocessing of MD simulations, pyPolyBuilder may easily be incorporated in already existing workflows. Moreover, it is an open-source software and thus free of charge and of extensible functionality. Currently, the output formats are compatible with GROMOS and GROMACS simulation packages, but other formats may be implemented flexibly.

Acknowledgement

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement no. 764958. This was supported by NACAD/COPPE/UFRJ and partially financed by "Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil" (CAPES) - Finance Code 001. FAPERJ and CNPq are also acknowledged.

Supporting Information Available

Methodological details are available as Supporting Information.

References

(1) Lombardo, S. M.; Schneider, M.; Türeli, A. E.; Günday Türeli, N. Key for crossing the BBB with nanoparticles: the rational design. *Beilstein Journal of Nanotechnology* **2020**, *11*, 866–883.

(2) Khan, I.; Saeed, K.; Khan, I. Nanoparticles: Properties, applications and toxicities. *Arabian Journal of Chemistry* **2019**, *12*, 908–931.

(3) Jeevanandam, J.; Barhoum, A.; Chan, Y. S.; Dufresne, A.; Danquah, M. K. Review on nanoparticles and nanostructured materials: history, sources, toxicity and regulations. *Beilstein Journal of Nanotechnology* **2018**, *9*, 1050–1074.

(4) Kaur, D.; Jain, K.; Mehra, N. K.; Kesharwani, P.; Jain, N. K. A review on comparative study of PPI and PAMAM dendrimers. *Journal of Nanoparticle Research* **2016**, *18*, 146.

(5) Jain, K.; Verma, A. K.; Mishra, P. R.; Jain, N. K. Surface-Engineered Dendrimeric Nanoconjugates for Macrophage-Targeted Delivery of Amphotericin B: Formulation Development and In Vitro and In Vivo Evaluation. *Antimicrobial Agents and Chemotherapy* **2015**, *59*, 2479–2487.

(6) Dünweg, B.; Kremer, K. Molecular dynamics simulation of a polymer chain in solution. *The Journal of Chemical Physics* **1993**, *99*, 6983–6997.

(7) Perilla, J. R.; Goh, B. C.; Cassidy, C. K.; Liu, B.; Bernardi, R. C.; Rudack, T.; Yu, H.; Wu, Z.; Schulten, K. Molecular dynamics simulations of large macromolecular complexes. *Current Opinion in Structural Biology* **2015**, *31*, 64–74.

(8) Gartner, T. E.; Jayaraman, A. Modeling and Simulations of Polymers: A Roadmap. *Macromolecules* **2019**, *52*, 755–786.

(9) Vögele, M.; Köfinger, J.; Hummer, G. Hydrodynamics of Diffusion in Lipid Membrane Simulations. *Physical Review Letters* **2018**, *120*, 268104.

(10) Eichenberger, A. P.; Allison, J. R.; Dolenc, J.; Geerke, D. P.; Horta, B. A. C.; Meier, K.; Oostenbrink, C.; Schmid, N.; Steiner, D.; Wang, D.; Van Gunsteren, W. F. GROMOS++ software for the analysis of biomolecular simulation trajectories. *Journal of Chemical Theory and Computation* **2011**, *7*, 3379–3390.

(11) Kunz, A. P. E.; Allison, J. R.; Geerke, D. P.; Horta, B. A. C.; Hünenberger, P. H.; Riniker, S.; Schmid, N.; Van Gunsteren, W. F. New functionalities in the GROMOS biomolecular simulation software. *Journal of Computational Chemistry* **2012**, *33*, 340–353.

(12) Abraham, M. J.; Murtola, T.; Schulz, R.; Páll, S.; Smith, J. C.; Hess, B.; Lindahl, E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* **2015**, *1-2*, 19–25.

(13) Case, D. A.; Cheatham, T. E.; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J. The Amber biomolecular simulation programs. *Journal of Computational Chemistry* **2005**, *26*, 1668–1688.

(14) Tomalia, D. A.; Naylor, A. M.; Goddard, W. A. Starburst Dendrimers: Molecular-Level Control of Size, Shape, Surface Chemistry, Topology, and Flexibility from Atoms to Macroscopic Matter. *Angewandte Chemie International Edition in English* **1990**, *29*, 138–175.

(15) Ramos, M. C.; Horta, V. A. C.; Horta, B. A. C. Molecular Dynamics Simulations of PAMAM and PPI Dendrimers Using the GROMOS-Compatible 2016H66 Force Field. *Journal of Chemical Information and Modeling* **2019**, *59*, 1444–1457.

(16) Dykes, G. M. Dendrimers: A review of their appeal and applications. *Journal of Chemical Technology and Biotechnology* **2001**, *76*, 903–918.

(17) SVENSON, S.; TOMALIA, D. Dendrimers in biomedical applications—reflections on the field. *Advanced Drug Delivery Reviews* **2005**, *57*, 2106–2129.

(18) Jikei, M.; Kakimoto, M.-a. Hyperbranched polymers: a promising new class of materials. *Progress in Polymer Science* **2001**, *26*, 1233–1285.

(19) Gao, C.; Yan, D. Hyperbranched polymers: from synthesis to applications. *Progress in Polymer Science* **2004**, *29*, 183–275.

(20) Kesharwani, P.; Jain, K.; Jain, N. K. Dendrimer as nanocarrier for drug delivery. *Progress in Polymer Science* **2014**, *39*, 268–307.

(21) Cheng, Y.; Zhao, L.; Li, Y.; Xu, T. Design of biocompatible dendrimers for cancer diagnosis and therapy: current status and future perspectives. *Chemical Society Reviews* **2011**, *40*, 2673.

(22) Yang, J.; Zhang, Q.; Chang, H.; Cheng, Y. Surface-Engineered Dendrimers in Gene Delivery. *Chemical Reviews* **2015**, *115*, 5274–5300.

(23) Van Aalten, D. M. F. PRODRG, a program for generating molecular topologies and unique molecular descriptors from coordinates of small molecules. *Journal of Computer-Aided Molecular Design* **1996**, *10*, 255–262.

(24) Schüttelkopf, A. W.; Van Aalten, D. M. F. PRODRG: A tool for high-throughput crystallography of protein-ligand complexes. *Acta Crystallographica Section D: Biological Crystallography* **2004**, *60*, 1355–1363.

(25) Widmann, A. H. Simulation of the intrinsic viscosity of hyperbranched polymers with varying topology. 1. Dendritic polymers built by sequential addition. *Computational and Theoretical Polymer Science* **1998**, *8*, 191–199.

(26) Faulon, J. L. Stochastic generator of chemical structure. 4. Building polymeric systems with specified properties. *Journal of Computational Chemistry* **2001**, *22*, 580–590.

(27) Wang, J.; Wang, W.; Kollman, P. a.; Case, D. a. Antechamber, An Accessory Software Package For Molecular Mechanical Calculations. *J. Am. Chem. Soc* **2001**, *222*, U403.

(28) Ogawa, H. GBstudio: A builder software on periodic models of CSL boundaries for molecular simulation. *Materials Transactions* **2006**, *47*, 2706–2710.

(29) Jo, S.; Kim, T.; Im, W. Automated builder and database of protein/membrane complexes for molecular dynamics simulations. *PLoS ONE* **2007**, *2*.

(30) Ribeiro, A. A. S. T.; Horta, B. A. C.; De Alencastro, R. B. MKTOP: A program for automatic construction of molecular topologies. *Journal of the Brazilian Chemical Society* **2008**, *19*, 1433–1435.

(31) Malde, A. K.; Zuo, L.; Breeze, M.; Stroet, M.; Poger, D.; Nair, P. C.; Oostenbrink, C.; Mark, A. E. An Automated Force Field Topology Builder (ATB) and Repository: Version 1.0. *Journal of Chemical Theory and Computation* **2011**, *7*, 4026–4037.

(32) Koziara, K. B.; Stroet, M.; Malde, A. K.; Mark, A. E. Testing and validation of the Automated Topology Builder (ATB) version 2.0: Prediction of hydration free enthalpies. *Journal of Computer-Aided Molecular Design* **2014**, *28*, 221–233.

(33) Stroet, M.; Caron, B.; Visscher, K. M.; Geerke, D. P.; Malde, A. K.; Mark, A. E. Automated Topology Builder Version 3.0: Prediction of Solvation Free Enthalpies in Water and Hexane. *Journal of Chemical Theory and Computation* **2018**, *14*, 5834–5845.

(34) Gomes, T. C. F.; Skaf, M. S. Cellulose-builder: A toolkit for building crystalline structures of cellulose. *Journal of Computational Chemistry* **2012**, *33*, 1338–1346.

(35) Schmidt, T. H.; Kandt, C. LAMBADA and InflateGRO2: Efficient membrane alignment and insertion of membrane proteins for molecular dynamics simulations. *Journal of Chemical Information and Modeling* **2012**, *52*, 2657–2669.

(36) Bellini, R. G.; Guimarães, A. P.; Pacheco, M. A. C.; Dias, D. M.; Furtado, V. R.; De Alencastro, R. B.; Horta, B. A. C. Association of the anti-tuberculosis drug rifampicin with a PAMAM dendrimer. *Journal of Molecular Graphics and Modelling* **2015**, *60*, 34–42.

(37) Goodman, E. D. Introduction to Genetic Algorithms. Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation. New York, NY, USA, 2014; p 205–226.

(38) Kora, P.; Yadlapalli, P. Crossover Operators in Genetic Algorithms: A Review. *International Journal of Computer Applications* **2017**, *162*, 34–36.

(39) Guo, J.; Shi, K. To Preserve or Not to Preserve Invalid Solutions in Search-Based Software Engineering: A Case Study in Software Product Lines. Proceedings of the 40th International Conference on Software Engineering. New York, NY, USA, 2018; p 1027–1038.

(40) Lambora, A.; Gupta, K.; Chopra, K. Genetic Algorithm- A Literature Review. 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). 2019; pp 380–384.

(41) Heskins, M.; Guillet, J. E. Solution Properties of Poly(N-isopropylacrylamide). *Journal of Macromolecular Science: Part A - Chemistry* **1968**, *2*, 1441–1455.

(42) Walter, J.; Ermatchkov, V.; Vrabec, J.; Hasse, H. Molecular dynamics and experimental study of conformation change of poly(N-isopropylacrylamide) hydrogels in water. *Fluid Phase Equilibria* **2010**, *296*, 164–172.

(43) Deshmukh, S. A.; Sankaranarayanan, S. K. R. S.; Suthar, K.; Mancini, D. C. Role of Solvation Dynamics and Local Ordering of Water in Inducing Conformational Transitions in Poly(N -isopropylacrylamide) Oligomers through the LCST. *The Journal of Physical Chemistry B* **2012**, *116*, 2651–2663.

(44) Boţan, V.; Ustach, V.; Faller, R.; Leonhard, K. Direct Phase Equilibrium Simulations of NIPAM Oligomers in Water. *The Journal of Physical Chemistry B* **2016**, *120*, 3434–3440.

(45) Podewitz, M.; Wang, Y.; Quoika, P. K.; Loeffler, J. R.; Schauperl, M.; Liedl, K. R. Coil–Globule Transition Thermodynamics of Poly(N -isopropylacrylamide). *The Journal of Physical Chemistry B* **2019**, *123*, 8838–8847.

(46) Quoika, P. K.; Podewitz, M.; Wang, Y.; Kamenik, A. S.; Loeffler, J. R.; Liedl, K. R. Thermosensitive Hydration of Four Acrylamide-Based Polymers in Coil and Globule Conformations. *The Journal of Physical Chemistry B* **2020**, *124*, 9745–9756.

(47) Horta, B. A. C.; Merz, P. T.; Fuchs, P. F. J.; Dolenc, J.; Riniker, S.; Hünenberger, P. H. A GROMOS-Compatible Force Field for Small Organic Molecules in the Condensed Phase: The 2016H66 Parameter Set. *Journal of Chemical Theory and Computation* **2016**, *12*, 3825–3850, PMID: 27248705.

(48) Maiti, P. K.; Çağin, T.; Wang, G.; Goddard, W. A. Structure of PAMAM dendrimers: Generations 1 through 11. *Macromolecules* **2004**, *37*, 6236–6254.

(49) Idris, A. O.; Mamba, B.; Feleni, U. Poly (propylene imine) dendrimer: A potential nanomaterial for electrochemical application. *Materials Chemistry and Physics* **2020**, *244*, 122641.

(50) Feng, X.; Taton, D.; Ibarboure, E.; Chaikof, E. L.; Gnanou, Y. Janus-Type Dendrimer-like Poly(ethylene oxide)s. *Journal of the American Chemical Society* **2008**, *130*, 11662–11676.

(51) Wang, L.; Shi, C.; Wang, X.; Guo, D.; Duncan, T. M.; Luo, J. Zwitterionic Janus Dendrimer with distinct functional disparity for enhanced protein delivery. *Biomaterials* **2019**, *215*, 119233.

(52) Suk, J. S.; Xu, Q.; Kim, N.; Hanes, J.; Ensign, L. M. PEGylation as a strategy for improving nanoparticle-based drug and gene delivery. *Advanced Drug Delivery Reviews* **2016**, *99*, 28–51.

(53) Luong, D.; Kesharwani, P.; Deshmukh, R.; Mohd Amin, M. C. I.; Gupta, U.; Greish, K.; Iyer, A. K. PEGylated PAMAM dendrimers: Enhancing efficacy and mitigating toxicity for effective anticancer drug and gene delivery. *Acta Biomaterialia* **2016**, *43*, 14–29.