

VDNA-Lab: A Computational Simulation Platform for DNA Multi-strand Dynamics

Frankie Spencer¹^a, Usman Sanwal¹^b and Eugen Czeizler^{1,2}^c

¹*Department of Information Technologies, Åbo Akademi University, Turku, Finland*

²*National Institute for Research and Development of Biological Sciences, Bucharest, Romania*

Keywords: DNA Assembly Simulator, Multi-strand Assembly, Rule-based Modeling, Graphical User Interface, Course-grained Modeling.

Abstract: The dynamics of nucleic-acids dynamical systems is intrinsically based on local interaction. The major acting mechanisms are that of Watson-Crick complementarity on one-hand, generating binding events, and thermal energy on the other, generating random motion and un-binding. It is thus predictable that such systems can be successfully captured by computational modeling paradigms based on local interactions, such as the rule-based modeling methodology. In this research we introduce the Virtual DNA Lab (VDNA-Lab) a simulation tool which provides an easy to use graphical interface for creating, running and visualizing synthetic simulations for DNA assembly systems, such as assembly of DNA nanostructures, strand displacement cascades systems, DNA-tile assembly etc. It employs a custom designed model, implemented in the BioNetGen Language (BNGL) formalism, to capture the DNA dynamics, as well as the NFsim computational modeling engine to run simulations and generate outputs. These outputs can be visualized using the VDNA-Lab's own visualization tool, which allows also for further analysis and filtering. The software is freely available at https://github.com/Frankie-Spencer/virtual_dna_lab.


1 INTRODUCTION


The fast increase of computational power experienced in the past decade has made it possible for computer scientists to develop various modeling and simulation techniques giving them access to very high levels of details for various self-assembly phenomena, despite that achieving such levels is prohibitively expensive in real experiments. Thus, the use of computer simulations in the field of self-assembly is now well accepted as well as its role in better understanding of many quantitative details of such systems (Thomas and Schwartz, 2017), (Kamerlin and Warshel, 2011), (Glotzer et al., 2004). A major challenge for the modeling of self-assembly biological phenomena lies in the fact that we have to deal with an extremely large space of possible assembly pathways accessible to the intermediate species of a self-assembly reaction network. Another challenge arises from the fact that self-assembly reactions are sensitive to many physical constraints (e.g. structural, chemical) available within


the original in vivo environment, conditions which are hard to be replicated within the in silico computational models (Ariga and et al., 2008). Despite these challenges there are a variety of modeling methods available such as SSA/Gillespie approaches (Sweeney and et al., 2008), (Blinov and et al., 2004) or Rule-based approaches (Mohammed et al., 2017), (Danos and et al., 2009), which have been proven valuable for the study and analysis of self-assembly systems.

Nanotechnology is one of the leading worldwide undertakings of the scientific community. Within this field, DNA has proved itself as a very powerful and versatile construction material (Kuzyk et al., 2009), (Li and et al., 2011) and (Lund and et al., 2010). Recent advances in DNA-based nanotechnology have opened the way towards the systematic engineering of inexpensive nanoscale devices for a multitude of purposes (Krishnan and Simmel, 2011), (Li and et al., 2011), (Lund and et al., 2010). Self-assembly properties of DNA, which are based on specific and predictable Watson-Crick based pairing rules, make it a highly malleable and controllable nanomaterial (Seeman, 2003).

There are several tools available for modeling and

^a  <https://orcid.org/0000-0002-6775-1832>

^b  <https://orcid.org/0000-0002-2178-3329>

^c  <https://orcid.org/0000-0002-1607-1554>

visualization of complex DNA-based self-assembly constructions. These tools include the Visual DSD (DNA Strand Displacement) tool (Lakin and et al., 2011), focused on modeling DNA strand displacement reaction systems, Iowa State University Tile Assembly Simulator (ISU TAS) (Patitz, 2011), focused on modeling DNA-tile assembly, or oxDNA (Poppleton and et al., 2020), which is a coarse grained modeling tool for the physical atom-level simulation of DNA assembly systems. Rule-based modeling approaches have also been used for modeling the kinetics of DNA-based Tile Assembly Systems (Mohammed et al., 2017), as well as for DNA strand displacement cascades (Gautam. et al., 2020).

In our work, we provide a framework for rule-based modeling of generalized DNA multi-strand dynamics, where DNA molecules and their interactions are modeled at nucleotide level. We have used the BioNetGen Language (BNGL) formalism (Blinov and et al., 2004) and the NFsim computational platform (Sneddon and et al., 2011) as well as a comprehensive collection of aiding (Python) subroutines which facilitate the user interface and make the modeling experience available for the user without having any prior knowledge of the BNGL modeling framework. In order to track and report global mapping of the components within a heterogeneous complex, we have developed a visualization interface, details provided in Section 3, which not only helps to visualize the results but also can be used for further numerical analysis. The software is freely available, see (Spencer et al., 2021).

2 MATERIAL AND METHODS

2.1 Rule-based Modeling

Modeling and simulation of self-assembly biochemical systems is a challenging computational task, as the modeler has to deal with both a high combinatorial number of species, namely all the generated partial- and sub-assemblies, and also with a large numbers of reactions between these species. The Rule-based modeling approach, which employs the idea of subdividing molecules, or generally species-entities, into their primary components, denoting protein domains, active sites or any other feature of the particle, can be used to solve this problem (Faeder et al., 2009). Within the rule-based modeling paradigm, molecules are represented as *agents* with a finite number of independent *sites*. The sites provide the agent-agent binding functionality and as a result molecular complexes are generated. *Rules* are only defined on the basis of

local *patterns* which provide a compact representation about agents' interaction. In rule-based modeling, models are defined in terms of a specific syntax (rules) instead of writing the model mathematically directly. After that a semantic is presented in order to cover the differences between what is written and the mathematical definition of its computation. A carefully described syntax increases the model's accessibility for domain experts that are not familiar with the modeling and mathematical formalisms (Faeder et al., 2009).

Two useful simulation tools for rule-based modeling are BioNetGen (Faeder et al., 2009) and NFsim (Sneddon and et al., 2011). BioNetGen is a BNGL-compatible simulation tool that implements various indirect methods including both deterministic and stochastic methods. NFsim is also a BNGL-compatible simulation tool that implements a (stochastic) direct method which applies a specific network-free simulation and thus avoids enumeration of species and reactions, that may be intractable for large models (Yang and Hlavacek, 2011). There are six different blocks of a BNGL file, namely: *parameters*, *species*, *observables*, *molecule types*, *reaction rules* and *functions*. Further detail about these blocks and NFsim simulator can be found in (Faeder et al., 2009) and (Sneddon and et al., 2011).

2.2 A Computational Modeling Framework for DNA Multi-strand Dynamics

We have created a generic computational modeling framework for DNA multi-strand dynamics using the BioNetGen Language (BNGL) formalism and the NFsim computational platform. The DNA molecules and the subsequent interactions are captured starting from nucleotide level. Each nucleotide (as a bio-entity) is represented as an individual instance of a generic agent N (from nucleotide) which has 1 state-characteristic site, b (from base), and 3 connecting sites: 5', 3', and W. Relating to its biological counterpart, the site b can be in exactly one of the four possible states: A (Adenine), G (Guanine), C (Cytosine), or T (Thymine); moreover, once initialized, site b will never change its state. The remaining sites of agent N are used for connecting the nucleotide within the single-stranded molecule it is part of¹, i.e., us-

¹DNA is composed primarily of nucleotides with a sugar-phosphate backbone; the nucleotides are attached to the backbone to form a structure which is known as a *single-stranded DNA* (ssDNA) molecule. The ssDNA is represented with two different ends. By convention, one end of the backbone is known as the 5' end and the second one is

ing sites 5' and 3', and for connecting the nucleotide with a complementary base-pair², using site W. For example, within our rule-based model, the basic structure for the single-stranded DNA (ssDNA) molecule ATTGCTA is shown in Figure 1.

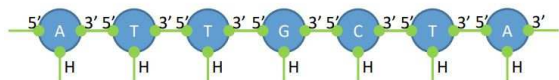


Figure 1: The schematic representation of the ssDNA complex ATTGCTA within our BNGL model.

VDNA-Lab takes as input populations of isolated (5'-to-3' oriented) single-stranded DNA molecules (ssDNAs) and simulates their binding and dissociation reactions. At the core of this model implementation lie 12 binding and un-binding local interaction rules, each with its own kinetic rate constant, and each implemented through one or several rule-based reactions. Using these reactions/rules, we capture the dynamics of the DNA-dynamical system, by modeling:

- the initial binding of short toeholds, i.e., short complementary sequences of 3-to-7 nucleotides;
- the "breathing" dynamics in-between bounded ssDNAs, i.e. zipping and unzipping reactions;
- random un-binding events (between one pair of nucleotides);
- the un-binding of loosely connected ssDNAs.

The list of all rule-types and their associated kinetic rate constants is presented in Table 1. By default, VDNA-Lab provides some normalized values for these kinetic rate constants, see the 4-th column of the table. However, any of these values can be updated by the user using the *Advanced* option.

3 RESULTS: THE VDNA-Lab SOFTWARE

We have developed the VDNA-Lab (Virtual DNA Lab) software to provide an easy to use graphical interface for creating, running and visualizing synthetic simulations for DNA assembly systems, such as assembly of DNA nanostructures, strand displacement cascades systems, DNA-tile assembly etc. It uses the NFsim computational modeling engine to run simulations and generate outputs. These outputs can be visualized using the VDNA-Lab's visualization tool,

known as the 3' end.

²The base-pair nucleotides A-T and C-G are complementary which means their shape allows them to bond together with hydrogen bonds.

which allows also some further analysis of the simulation output. Here we explain the different features and parameters of the VDNA-Lab software.

The tool consists of three main functionalities, each reachable from a specific tab on the upper left corner of the main window:

- *Create Test Tube* for creating new test tube input files; these are called also species files, as they are stored on computer with the extension .species
- *Run Experiment* for simulating the assembly process within a given input test tube file, and
- *Visualize Test Tube* for visualizing the content of a test tube at the beginning, middle, or end of an experiment.

The *Create Test Tube* feature of VDNA-Lab, see Fig. 2 is used to create/edit new species files, to be used as input in the Run Experiment process. ssDNA sequences are entered by the user (only strings over A,T,C and G are allowed to be entered as input) in the *Desired ssDNA sequence* tab and the number of copies of the generated ssDNA structure has to be introduced in the *Amount* tab (by default 1 copy is created). Once the user presses the *Add* button, these sequences are added to the *List of ssDNA to be created* text box. After the ssDNA sequences are created, the user can *Edit*, *Delete*, or *Reset all* of them. The user can also import the ssDNA sequences, or even more complicated multi-ssDNA complexes, using the *Import from Species* button on the right hand side. Such species files can be extracted for example from the default output files generated during the numerical simulation phase, i.e., from the "Run experiment" tab. Thus, the user can select the input

³single complementary bases

⁴The two interacting ssDNA segments can be part of the same ssDNA molecule, i.e., a hairpin loop, or be two distinct ssDNA molecules bound within the same complex.

⁵That is, both ssDNA strands continue with pairwise non-bounded sequences to one direction -at least one strand should not be bound to anything else- and bound sequences of length at least 1 to the other direction

⁶There are several cases within the model where two nucleotides become/remain bounded, although such a situation might not occur experimentally. These situations are: i) single complementary (or non-bound) ssDNA sequences; ii) single complementary base positioned at the end of a non-complementary (or not-bound) sequence; iii) single complementary base, where the complement is on the same ssDNA, at distance 0 or 1; iv) single complementary base when the opposite neighbors of the pair are bound but not complementary, i.e., are bound but not to each-other

⁷The default value of the kinetic rate constant K_{max} implementing an instantaneous reaction is set to 10^5

Table 1: The list of 12 rules governing the dynamics of the DNA self-assembly process, and the default values of their associated kinetic rate constants. The kinetic rate constants are scaled according to the rate of the intra-complex toehold binding of complementary segments, i.e., Rule 4, which is normalized to value 1.

Rule#	Description of rule action	kinetic param. (k.p.)	default val. of k.p.
Rule 1	toehold binding of compl. seq. (3-to-7 bases) of 2 un-connected ssDNA	k_1	0.001
Rule 2	binding of s.c.b. ¹ of two immediately-connected ssDNA segments ²	k_2	300
Rule 3	binding of s.c.b. of two closely-connected (1-off neighbor connection) ssDNA segments ²	k_3	30
Rule 4	toehold binding of compl. seq. (3-to-7 bases) of two connected ssDNA	k_4	1
Rule 5	un-binding of s.c.b. positioned at a split ³	k_5	30
Rule 6	un-binding of s.c.b. positioned in the middle of a compl. sequence	k_6	0.1
Rule 7	un-binding of s.c.b. where at least one ssDNA ends on that position	k_7	50
Rule 8	very rapid (instantaneous) un-binding between s.c.b. in abnormal conditions ⁴	k_8	K_{max} ⁵
Rule 9	random unbinding of a pair of bound nucleotides	k_9	0.01
Rule 10	rapid un-binding of a size-2 comp. seq. in the middle of non-compl./non-bound seq.	k_{10}	300
Rule 11	rapid un-binding of a size-3 comp. seq. in the middle of non-compl./non-bound seq.	k_{11}	100
Rule 12	rapid un-binding of a size-4 comp. seq. in the middle of non-compl./non-bound seq.	k_{12}	10

sequences of an experiment to be the output multi-ssDNA structures generated during previous simulations. Moreover, the newly generated input file can be saved as a new species file, using the "Save as custom species file" check-box, or it will become (by default) the current test tube to be simulated.

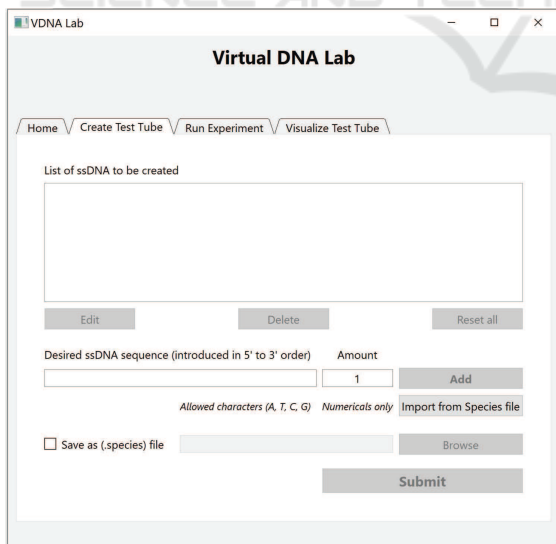


Figure 2: The *Create Test Tube* environment for creating/editing test tube (.species) files.

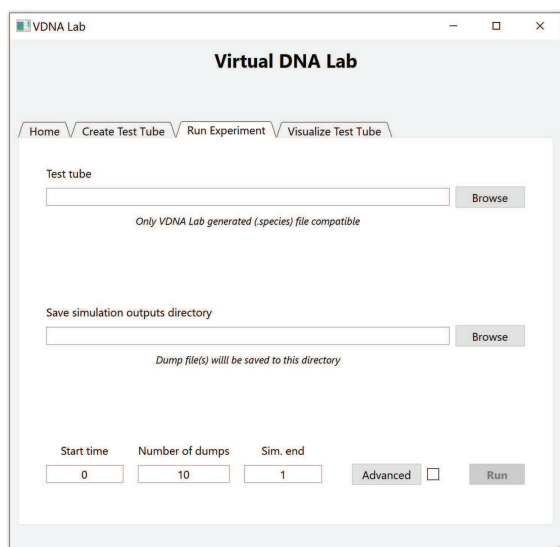
The *Run Experiment* feature, see Fig. 3 is used to numerically simulate the DNA assembly process from

a given input .species file, or (by default) from the current test tube. The VDNA-Lab software uses NFsim simulator on background to run experiments. NFsim needs several parameters to run and produce output (.dump) files for a certain simulation. The user can enter numerical values for the *Start time*, *Number of dumps* and *Sim. end* (i.e., end model-time for the simulation); the default values for these entries are 0, 10, and 1, respectively. The output, i.e., .dump, files are saved within a newly created folder, named after the name of the input file and the time of the simulation; the location of this folder is introduced by the user using the *Save simulation output directory* tab.

The *Run Experiment* feature has also several *Advanced* options. By selecting this tab, a new window opens giving the user advanced choices regarding three modeling aspects, each contained within a delimited area of the window:

First, the user can select between using the default BNGL model file⁸, describing the entire model's entities and all their interaction rules, or a separate user-specified .bngl model file, possibly describing a user-altered variant of this model; such possible model-variant modifications are suggested with respect to the custom temperature-controlled annealing protocols detailed below.

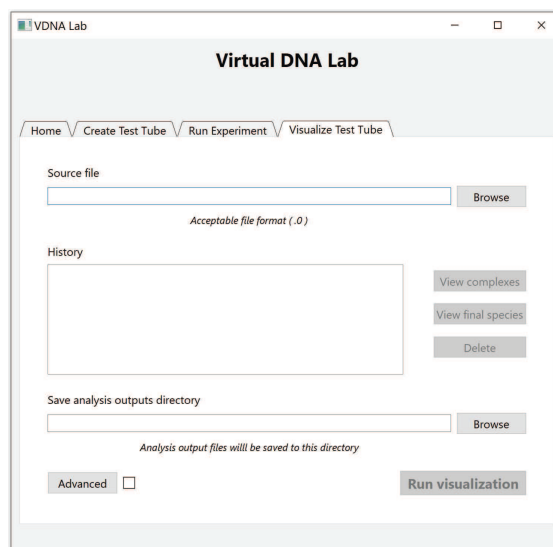
⁸The default BNGL model file is called master_VDNA-LAB.BNGL_model.bngl, and it is placed within the folder ... \system_files\reference_files of the VDNA-Lab installation folder

Figure 3: The *Run simulation* environment.

The second advanced feature offers the possibility to modify the kinetic rate constants of the 12 reaction rules indicated in Table 1, which are governing the entire assembly (and dis-assembly) dynamics.

Finally, the third advanced feature allows the user to define an annealing protocol for the simulation/experiment, during which the temperature of the system, and subsequently part of the reaction rates, are modified several times mid simulation. Namely, the user can specify the start and end temperature of the system, the (constant) temperature decrease/increase per annealing step, as well as the length of time each annealing step will last. By default, without the annealing mechanism, the system is assumed at a (normalized) temperature $Temp = 1$. When the annealing mechanism is activated, the kinetic rate constants of reaction rules 5–12, from within Table 1, are multiplied by the $Temp$ values, while the kinetic rate constants of reaction rules 1–4 remain unmodified. If the user wants to create new $Temp$ -dependable kinetic rate constants for any of the above rules, then he can modify the default BNGL model file⁸ within the "begin functions ... end functions" block, and then select the option of using this custom .bnl model file.

The *Visualize Test Tube* window, see Fig. 4, can be used by the user to generate a browser-based text-enhanced visualization of the content of a simulation output (.dump) file, see e.g. the visualization from Fig. 5 associated to the running example discussed in the next section. Within the visualization, each complex, composed from one or several bound ssDNA, is displayed separately, with each ssDNA within the complex on a separate line. Per each ssDNA we rep-

Figure 4: The *Visualized Test Tube* environment.

resent the nucleotide sequence, either on the 5' to 3' order, or on the reverse order. The first ssDNA of a complex is always displayed on the 5' to 3' order, while subsequent ssDNAs within the complex have the reversed order of the first complemented sequence. Moreover, if some particular nucleotide is bound to another, by hovering the mouse over this nucleotide its complement will highlight too. Also, consecutive segments of bound nucleotides will have a similar coloring. Due to these features, the user can easily see the active domains of the complex, i.e., sequences of consecutive nucleotides bound to complementary contiguous sequences.

The *Advanced* option gives user further ability to obtain filtered results, highlights, and also get stats about the selected filtered complexes. Further details made available by the visualization feature will be described below, when we provide a running example of our software on several simple input instances.

3.1 Running Examples

In this section, we present two running examples for our computational simulation platform, one designed to capture a DNA strand-displacement reaction, and the other following the assembly of DAE-E DNAtiles (Jiang and et al., 2017). In both cases, VDNA-Lab takes as input populations of isolated (or partially bounded) oriented ssDNAs, and simulates their binding and dissociation reactions. The kinetic rate constants used within these examples are the software default values, which means that the rates are normalized and thus the current model-time is different from what one might observe in a real wet-lab experiment.

In our first test case, for the first simulation, we introduced 20 copies of the ssDNAs:

ACGATTGCT and *CCAGATCGT*

We simulated the assembly for 0.1 second (model-time), and we visualize the content of the test-tube, see Fig. 5 a) for a fraction of this visualization⁹.

On top of the visualization window, we display some basic statistics providing the content of the test-tube, namely the average, minimum and maximum number of ssDNAs per complex. For example, for our test-tube visualization in Fig. 5 a) we have that the average number of ssDNAs per complex is 1.33, while the minimum and maximum numbers of ssDNA within a complex are 1 and 2, respectively.

Within a visualization window, each distinct configuration corresponding to a complex is visualized only once. However, within a given test-tube (i.e., configuration) there could be many identical copies of the same complex. In such a case, the amount of identical copies are summed up and mentioned accordingly on top of each complex, as well as the number of ssDNAs that are included in that complex. For example, regarding the first visualized complex from Fig. 5 a) we can say that within that test-tube there are 9 complexes having the exactly same configuration. Single ssDNA within a complex can be identified as a chain of nucleotides placed on consecutive position. Multiple ssDNAs within a complex appear each on a separate line. The nucleotides that are bound to a complementary pair appear in color, whereas non-complemented nucleotides are displayed in gray. Minor color shift can be noticed within nucleotides that are consecutively complemented or that are closely located in an ssDNA.

Continuing our example, after the initial assembly, using the *Import from Species* function from the *Create Test Tube* window we add to the current configuration 20 copies of the new ssDNA

AGCAATCGT

The simulation is restarted for another 0.5 seconds, after which the output is visualized in Fig. 5 b)¹⁰.

As it can be observed from Fig. 5 b), one can observe both the situations where a strand displacement is in process, for those complexes containing 3 ssDNA molecules, and situations in which one of the initially bounded ssDNA molecules has been completely displaced, and only the two fully complementary strands are either partial or completely bound.

⁹The entire content of the test tube and thus of the visualization window requires more space, and thus was omitted
¹⁰As before, due to space limitation only a fraction of the visualization window is displayed

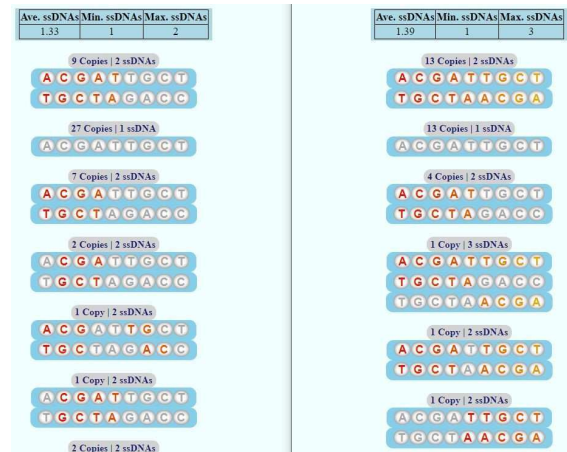


Figure 5: Fraction of simulation output after 0.1s and subsequent 0.5s after addition of new ssDNA.



Figure 6: Generated output of a DNA self-assembly system simulation and the desired layout of a DAE-E DNA-tile.

In a completely different experiment we have created a test-tube containing 20 copies of each of the 5 ssDNA molecules forming a DAE-E DNA-tile (Jiang and et al., 2017). We have simulated their assembly for 1 second and then visualized the output. In Fig. 6 we display one of the visualized complexes consisting of 5 ssDNA bound together into a successfully assembled DAE-E tile, as well as the desired schematic representation of such a tile.

4 CONCLUSIONS

In this paper, we have introduced VDNA-Lab (Spencer et al., 2021), a computational modeling framework for DNA multi-strand dynamics using the BioNetGen Language (BNGL) formalism (Harris, 2015) and the NFsim computational platform (Sneddon and et al., 2011). The ssDNAs are provided as input while the system simulates their binding and dissociation reactions. The model implementation is done with the help of 12 binding and un-binding local interaction rules. Each of these rules has its own, user-adjustable, kinetic rate constant and each of them is implemented through one or several rule-based reactions. Tracking and reporting the global mapping of the components within a heterogeneous complex is done with the help of a VDNA-Lab visualization subroutine. The content of the dynamical system is unloaded at various time-points within the

simulation, and it is re-assembled for visualization and further numerical analysis. A simple 2D graphical representation of the assembled complexes is also generated, where one can track the ssDNAs within the complex as well as all the binding interactions.

Different from other computational modeling frameworks for DNA strand assembly, we can modify some of the systems parameters, such as the temperature of the system, during the system simulation. Thus, we can model also an entire annealing process for the formation of a DNA structure. The underlining computational modeling engine used by VDNA-Lab, namely NFsim, allows also for other parameters to be adjusted mid-simulation: e.g., we can define specific binding/un-binding kinetic rates for each different length-3 subsequence. Currently, VDNA-Lab does not have these features implemented.

ACKNOWLEDGEMENTS

This work was partially supported by the Academy of Finland, project 311371/2017, and by the Romanian National Authority for Scientific Research and Innovation, PED grant 2391.

REFERENCES

- Ariga, K. and et al. (2008). Challenges and breakthroughs in recent research on self-assembly. *Science and Technology of Advanced Materials*, 9(1):014109.
- Blinov, M. L. and et al. (2004). BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, 20(17):3289–3291.
- Danos, V. and et al. (2009). Rule-based modelling and model perturbation. In *Transactions on Computational Systems Biology XI*, pages 116–137. Springer.
- Faeder, J. R., Blinov, M. L., and Hlavacek, W. S. (2009). *Rule-Based Modeling of Biochemical Systems with BioNetGen*, pages 113–167. Humana Press, NJ.
- Gautam, V., Long, S., and Orponen, P. (2020). Ruledsd: A rule-based modelling and simulation tool for dna strand displacement systems. In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 3: BIOINFORMATICS*, pages 158–167.
- Glotzer, S. C., Solomon, M. J., and Kotov, N. A. (2004). Self-assembly: From nanoscale to microscale colloids. *AIChE Journal*, 50(12):2978–2985.
- Harris, L. A. e. a. (2015). BioNetGen 2.2: Advances in Rule-Based Modeling. *arXiv e-prints*, page arXiv:1507.03572.
- Jiang, S. and et al. (2017). Understanding the elementary steps in dna tile-based self-assembly. *ACS Nano*, 11(9):9370–9381.
- Kamerlin, S. C. L. and Warshel, A. (2011). Multiscale modeling of biological functions. *Phys. Chem. Chem. Phys.*, 13:10401–10411.
- Krishnan, Y. and Simmel, F. C. (2011). Nucleic acid based molecular devices. *Angewandte Chemie International Edition*, 50(14):3124–3156.
- Kuzyk, A., Laitinen, K. T., and Törmä, P. (2009). DNA origami as a nanoscale template for protein assembly. *Nanotechnology*, 20(23):235305.
- Lakin, M. R. and et al. (2011). Visual dsd: a design and analysis tool for dna strand displacement systems. *Bioinformatics*, 27:3211–3213.
- Li, J. and et al. (2011). Self-assembled multivalent dna nanostructures for noninvasive intracellular delivery of immunostimulatory cpg oligonucleotides. *ACS Nano*, 5(11):8783–9.
- Lund, K. and et al. (2010). Molecular robots guided by prescriptive landscapes. *Nature*, 465(7295):206–209.
- Mohammed, A., Czeizler, E., and Czeizler, E. (2017). Computational modelling of the kinetic tile assembly model using a rule-based approach. *Theoretical Computer Science*, 701:203 – 215. At the intersection of computer science with biology, chemistry and physics - In Memory of Solomon Marcus.
- Patitz, M. J. (2011). Simulation of self-assembly in the abstract tile assembly model with ISU TAS. *CoRR*, abs/1101.5151.
- Poppleton, E. and et al. (2020). Design, optimization and analysis of large DNA and RNA nanostructure through interactive visualization, editing and molecular simulation. *Nucleic Acids Research*, 48(12):e72–e72.
- Seeman, N. (2003). Dna in a material world. *Nature Cell Biology*, 421(6921):427–431.
- Sneddon, M. and et al. (2011). Efficient modeling, simulation and coarse-graining of biological complexity with nfsim. *Nat Methods*, 8(2):177–183.
- Spencer, F., Sanwal, U., and Czeizler, E. (2021). VDNA-Lab. Available at https://github.com/Frankie-Spencer/virtual_dna_lab, version 1.0.
- Sweeney, B. and et al. (2008). Exploring the parameter space of complex self-assembly through virus capsid models. *Biophysical Journal*, 94(3):772 – 783.
- Thomas, M. and Schwartz, R. (2017). Quantitative computational models of molecular self-assembly in systems biology. *Physical Biology*, 14(3):035003.
- Yang, J. and Hlavacek, W. S. (2011). The efficiency of reactant site sampling in network-free simulation of rule-based models for biochemical systems. *Physical Biology*, 8(5):055009.