Bioinformatics Advance Access published May 7, 2015

Gener: A minimal programming module for chemical controllers based on DNA strand displacement

Ozan Kahramanoğulları [1,2] and Luca Cardelli [3,4]

[1] Department of Mathematics, University of Trento
[2] The Microsoft Research - University of Trento Centre for Computational and Systems Biology
[3] Microsoft Research Cambridge
[4] Department of Computer Science, University of Oxford

Associate Editor: Dr. Jonathan Wren

ABSTRACT

Summary: Gener is a development module for programming chemical controllers based on DNA strand displacement. Gener is developed with the aim of providing a simple interface that minimizes the opportunities for programming errors: Gener allows the user to test the computations of the DNA programs based on a simple two domain strand displacement algebra, the minimal available so far. The tool allows the user to perform stepwise computations with respect to the rules of the algebra as well as exhaustive search of the computation space with different options for exploration and visualization. Gener can be used in combination with existing tools, and in particular, its programs can be exported to Microsoft Research's DSD tool as well as to LaTeX.

Availability: Gener is available for download at the Cosbi website at http://www.cosbi.eu/research/prototypes/gener as a windows executable that can be run on Mac OS X and Linux by using Mono.

Contact: ozan@cosbi.eu

1 INTRODUCTION

One of the goals of synthetic biology is constructing information processing systems for controlling biochemical systems at the molecular level. Such an achievement would pave the way for applications, e.g., to smart therapeutic devices that are capable of sensing their environments Douglas *et al.* (2012); Amir *et al.* (2014). Within a broad spectrum, various technologies are being developed to address different aspects of this vision. Applications in DNA nano-technology aim at harnessing the complexity of biochemical dynamics to control active molecular devices in vivo Zhang and Seelig (2011). Technologies based on DNA strand displacement algebras, in particular, the double stranded architecture with nicks on one strand Phillips and Cardelli (2009) is proving to be effective also in wet lab implementations of formally designed experiments Chen *et al.* (2013).

The double stranded DNA strand displacement algebras perform computations as a result of the interactions between single and double stranded DNA structures: the single stranded structures act as signals that are processed by double stranded structures that act as gates. The mechanism with which the signals are processed by the gates is toehold mediated branch migration and strand displacement Yurke and Jr. (2003); Zhang and Winfree (2009). By using this machinery, one can program, e.g., systems of chemical reaction networks that operate at the molecular level Soloveichika *et al.* (2010); Dalchau *et al.* (2014). In this setting, a single chemical reaction step is emulated by a sequence of DNA-strand displacement operations. Because an increase in additional steps introduces more opportunities for errors in design, simpler schemes for designing these molecular programs become more favorable. In this respect, two domain DNA strand displacement scheme provides a good platform for developing molecular programs as it is minimal in design while being sufficiently expressive for describing chemical reaction networks that are of interest from the point of view of molecular programming Cardelli (2013); Lakin *et al.* (2013).

Gener is a programing module that implements the two domain DNA strand displacement algebra described in Cardelli (2013). With Gener, the user can write a two domain strand displacement program, and test its stepwise computations. The programs can be analyzed by exhaustive search of the computation space, and

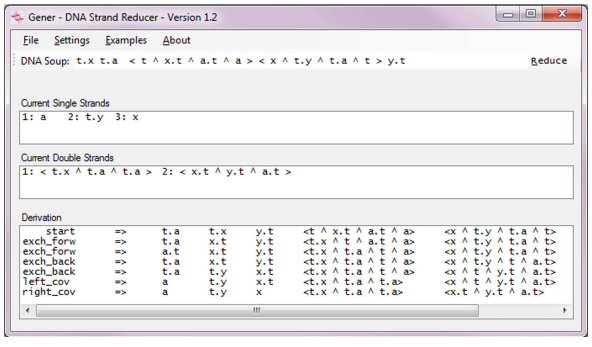

Fig. 1. A screen shot of a reduction performed on the built-in Example 1 that implements a transducer as in Cardelli (2013). The single strands consist of two domains, composed by '.'. One of the two composed domains can be a short domain, denoted with 't'. Any other string consisting of letters denotes a long domain. Complemented double strands are written in angle brackets '<' and '>', and they denote double strands consisting of strands and their Watson-Crick complements. We denote nicks on double strands with '^', which are the interruptions on one side of the double strands that make the interactions between the signals and gates possible.

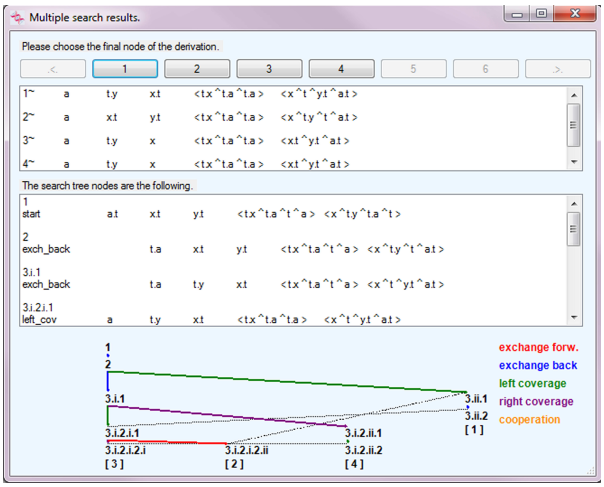

Fig. 2. A screen shot of a search performed on an example. For the search, 'all paths' option is chosen together with the display options 'search tree' and 'equal nodes'. The gray lines between the nodes denote the equal nodes.

the computations can be visualized in a tree representation with different options. Gener can be used in conjunction with Microsoft Research's DSD tool for simulation and analysis purposes as Gener programs can be exported to DSD Lakin *et al.* (2012, 2011), and computation traces can be exported to LaTeX for visualization. Gener contains introductory examples in its menu, which should be useful for a quick start. A manual is also available on the web.

2 METHODS

Gener programs consist of single and double stranded DNA structures, which are entered in the *DNA Soup* field at the top of the GUI. Gener aims at designing DNA displacement systems on the high abstract level, and uses the previously established notations (Figure 1). Gener can be used to observe the computations of the input DNA program. In the default setting, the user can choose from all the possible instances of the reduction rules by *reducing* the input, and proceed by applying the rules incrementally to observe a possible computation trace of the DNA soup. At each step the resulting strands and the computed derivation are displayed. An example derivation is shown in Figure 1. At each step during reduction, the user can perform a '*backtrack*' action by clicking the corresponding button of the GUI to return to the previous step.

Alternatively, the user can perform an exhaustive search of the computation space and choose from the available traces for displaying it. This is done by choosing the *search* option from the *settings* menu of the GUI. A search can be performed with further options: a simple search (without any further options being chosen) displays an enumeration of the available terminal computations of the strand structures by pruning the redundancies in the search space, and prompts a dialog window for the choice of the trace to be displayed. Choosing the *all paths* option for the search enables the enumeration of the computations with alternative paths, and the *variations* option includes to the enumeration also those paths with the same terminal as another path, but with an alternative trajectory.

Along with the search, the user can display the *search tree*, and with this, all the intermediate computations are listed in a reserved field. In the displayed search tree graph, different rules are denoted with different colors, which are listed in a legend. Choosing the *equal nodes* option includes a visualization of the nodes that result in the same DNA strand structures, which are connected with dashed gray lines in the search tree. The screen shot of a search tree visualization with an example is depicted in Figure 2.

By using the *file* menu, Gener DNA strand displacement programs can be exported to Microsoft Research's DSD language for simulation and analysis, and computation trajectories can be exported to LaTeX for typesetting.

3 DISCUSSION

Gener is developed for performing in silico experiments in design and debugging of DNA strand displacement systems. Potential applications include designing DNA sequences from desired DNA domain structures Zadeh *et al.* (2011). The search feature is useful for verification and analysis of reachable states of the designed systems. The sequences can then be ordered from IDT and executed in a basic wet lab with a fluorometer for reading the output Chen *et al.* (2013). The features of Gener are only limited by its minimalistic design, avoiding duplication of effort, and imagination. We thus foresee extensions as they will be required.

REFERENCES

Amir, Y., Ben-Ishay, E., Levner, D., Ittah, S., Abu-Horowitz, A., and Bachelet, I. (2014). Universal computing by dna origami robots in a living animal. *Nature Nanotechnology*, **9**, 353–357.

Cardelli, L. (2013). Two-domain DNA strand displacement. *Mathematical Structures in Computer Science*, **23**, 247–271.

Chen, Y.-J., Dalchau, N., Srinivas, N., Phillips, A., Cardelli, L., Soloveichik, D., and Seelig, G. (2013). Programmable chemical controllers made from DNA. *Nature Nanotechnology*, **8**, 755762.

Dalchau, N., Seelig, G., and Phillips, A. (2014). Computational design of reaction-diffusion patterns using DNA-based chemical reaction networks. *Int. Conf. on DNA Computing and Molecular Programming*, **8727**, 84–99.

Douglas, S. M., Bachelet, I., and Church, G. M. (2012). A logic-gated nanorobot for targeted transport of molecular payloads. *Science*, **335**(6070), 831–834.

Lakin, M., Parker, D., Cardelli, L., Kwiatkowska, M., and Phillips, A. (2012). Design and analysis of DNA strand displacement devices using probabilistic model checking. *J. R. Soc. Interface*, **9**(72), 1470–1485.

Lakin, M. R., Youssef, S., Polo, F., Emmott, S., and Phillips, A. (2011). Visual DSD: a design and analysis tool for DNA strand displacement systems. *Bioinformatics*, **27**(22), 3211–3213.

Lakin, M. R., Phillips, A., and Stefanovic, D. (2013). Modular verification of DNA strand displacement networks via serializability analysis. *Int. Conf. on DNA Computing and Molecular Programming*, **8141**, 133–146.

Phillips, A. and Cardelli, L. (2009). A programming language for composable dna circuits. *J. R. Soc. Interface*, **6**, 419–436.

Soloveichika, D., Seeliga, G., and Winfree, E. (2010). DNA as a universal substrate for chemical kinetics. *Proc. Nat. Acad. Sci.*, **107**(12), 5393–5398.

Yurke, B. and Jr., A. M. (2003). Using DNA to power nanostructures. *Genetic Programming and Evolvable Machines archive*, **4**(2), 111–122.

Zadeh, J., Wolfe, B., and Pierce, N. (2011). Nucleic acid sequence design via efficient ensemble defect optimization. *J Comput Chem*, **32**, 439–452.

Zhang, D. Y. and Seelig, G. (2011). Dynamic DNA nanotechnology using strand displacement reactions. *Nature Chem.*, **3**, 103–113.

Zhang, D. Y. and Winfree, E. (2009). Control of DNA strand displacement kinetics using toehold exchange. *J. Am. Chem. Soc.*, **131**(47), 17303–17314.