ACS Synthetic Biology

Technical Note

pubs.acs.org/synthbio

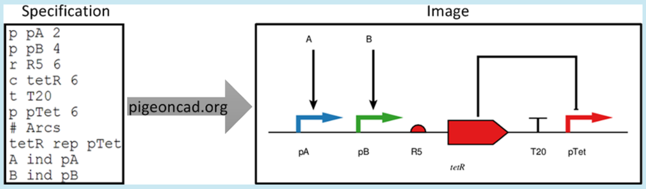
Pigeon: A Design Visualizer for Synthetic Biology

Swapnil Bhatia*,[†] and Douglas Densmore[†,‡]

[†]Department of Electrical and Computer Engineering, Boston University, Boston, Massachusetts, United States

[‡]Bioinformatics Department, Boston University, Boston, Massachusetts, United States

ABSTRACT: *Pigeon* is a Web-based tool that translates a textual description of a synthetic biology design into an image. It allows programmatic generation of design visualizations, is easy to learn, is easily extensible to new glyphs and notation, and can be connected to other software tools for visualizing their output. We present the *Pigeon* syntax, its current command set, and some examples of *Pigeon* programs and their output.



```
p pA 2
p pB 4
r R5 6
c tetR 6
t T20
p pTet 6
# Arcs
tetR rep pTet
A ind pA
B ind pB
```

KEYWORDS: *visualization, software*

■ MOTIVATION

As synthetic biology matures, it will need software tools to manage information complexity. Modeling and assembly tools apart, there is also a need for tools for communicating designs among synthetic biologists. Visual languages, syntax and semantics governing visual signs and their relationships, have proven to be useful in describing complex designs in many engineering disciplines. A visual sign system is especially useful in synthetic biology because it enables more effective communication and querying of sequence properties important to design and debugging such as adjacency, orientation, order, contiguity, and repetition, than nonvisual descriptions. A *de facto* visual sign system exists in synthetic biology and is heavily used for manually prototyping designs, and in describing their final form in publication. Such manual methods of visual description, however, are cumbersome, often incomplete or ambiguous, unscalable to larger designs and their many variants, variable across design communities, computer incomprehensible, and are inevitably repeated with generic illustration tools. To alleviate some of these shortcomings, we present a design visualization tool for synthetic biology called *Pigeon*.

■ WHAT IS *PIGEON*?

Pigeon is a software tool that generates a visualization of a synthetic biology design from its textual description. It is available for use in a Web browser: a designer enters a textual description of the design as per *Pigeon*'s syntax into a text box and submits it to the *Pigeon* server, and the server responds with a visualization of the design. *Pigeon* is accessible at the following URL: http://pigeoncad.org. The *Pigeon* design syntax allows for two types of specification: the glyphs in the design and the relationships among them, in that order. The glyphs in the design are specified by a sequence of lines of text of the form ⟨*command*⟩ ⟨*label*⟩ ⟨*color*⟩ nl, wherein only the first element is mandatory.

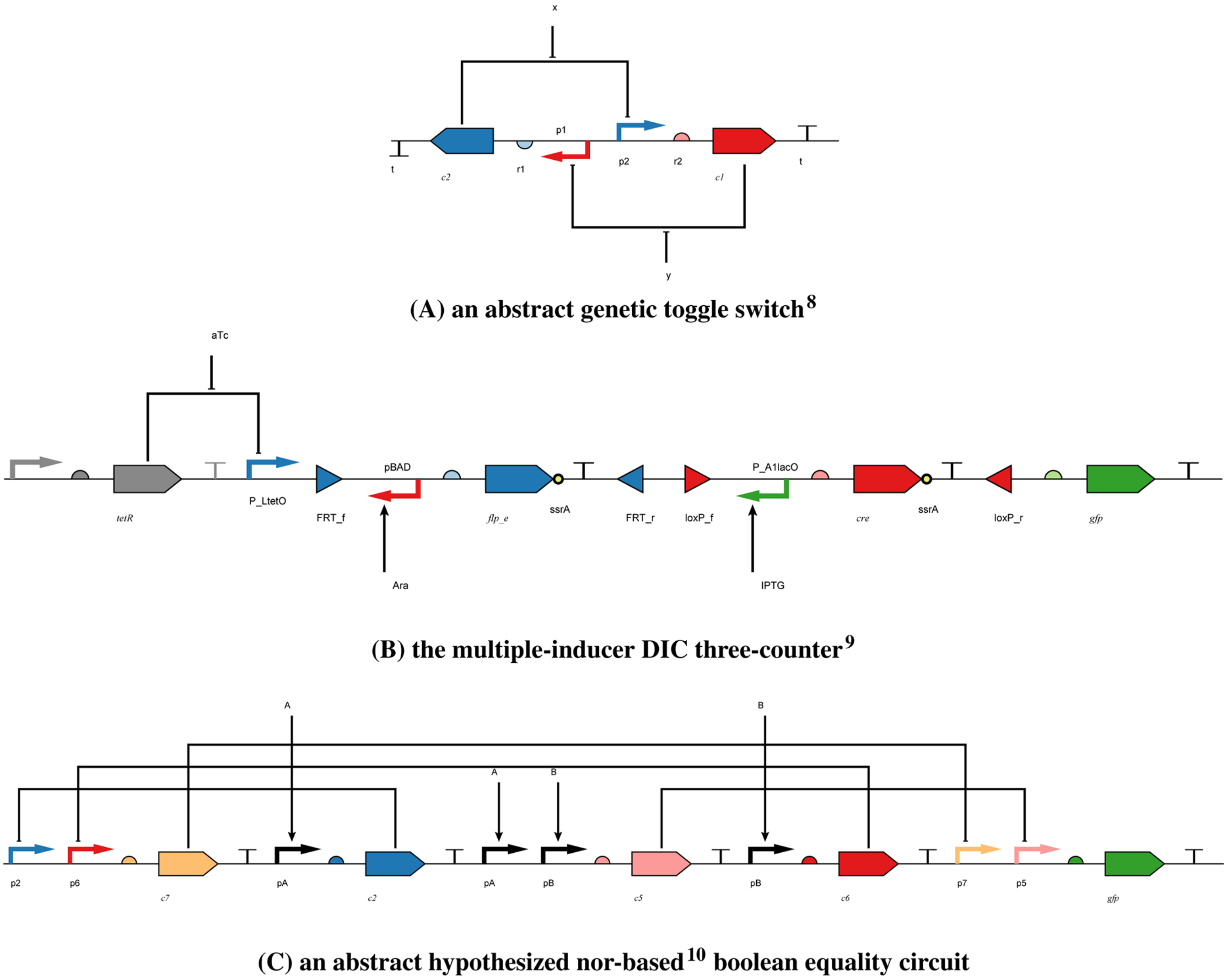
Commands. The current command set comprises the following commands: p, r, c, t, s, o, d, >, <, -, ., space, and l. In addition, the glyphs corresponding to p, r, c, and t can be inverted by prepending them with < like <p, <r, *etc.* The glyphs that some of these commands translate to can be seen from the examples in Figure 1 and Table 1. For example, the p command translates to a right-turn arrow glyph, the t command to a T glyph, and the > command to a right-pointing triangle glyph. A complete list is available on the *Pigeon* Web page. *Pigeon* uses glyphs commonly used by synthetic biologists and some from the SBOL compendium.[1]

Labels and Colors. A valid *label* is any string of characters from the alphabet {*a*, ..., *z*, *A*, ..., *Z*, 0, ..., 9, -, ., space}. *Pigeon* places the label near the glyph to allow annotating individual glyphs with text. The *color* is a positive integer between 1 and 14 representing one of a fixed set of colors. *Pigeon* currently supports a palette of 14 colors, 12 of which are suggested to be optimal for use with each other.[2] (We add black as a default color when the user leaves color unspecified, and the use of gray and black to allow the production of black-and-white visualizations.) The seven supported hues are blue, green, red, orange, purple, yellow, and black in that order. Odd numbered colors are lighter variants of the even numbered colors.[11]

Thus, c tetR 6 is an example of a valid glyph specification line in a design. Here, c is the command, tetR is an arbitrary label chosen by the user, and six is the color (dark red) chosen by the user for the glyph produced by the command c. A valid *Pigeon* specification line requires only the command to be present; the label and color are optional. When they are not specified, no label is produced, and the glyphs are colored black. If such a line is appended with the bigram nl, an abbreviation for "no label," then *Pigeon* will produce the glyph but without the label.

Relationships. *Pigeon* can also visualize relationships among glyphs using arcs. The arc specification follows the glyph specification and its beginning is marked with the line # Arcs. *Pigeon* supports two types of arcs: pointed and barred. The source and destination of each arc is specified using the labels

Received: March 9, 2013
Published: April 12, 2013



(A) an abstract genetic toggle switch[8]

(B) the multiple-inducer DIC three-counter[9]

(C) an abstract hypothesized nor-based[10] boolean equality circuit

Figure 1. Examples of design visualizations produced by *Pigeon* from the programs in Table 1.

of the glyphs given in the glyph specification. An arc line takes one of two forms: ⟨*source glyph name*⟩ ⟨*arc type*⟩ ⟨*destination glyph name*⟩, or ⟨*source glyph name*⟩ ⟨*arc type*⟩ ⟨*source glyph name-destination glyph name*⟩. Currently, source glyphs must be those specified by the c command, and destination glyphs must be those specified by the p command. The arc type must be ind or rep.

For example, if c TetR 6 and p pTet 6 are two lines in the glyph specification, then the line TetR rep pTet will draw a barred arc from any c glyph named TetR to any p glyph named pTet. Further, the line aTc rep TetR-pTet will draw a barred arc between the label aTc and the arc drawn in the previous example.

■ WHY *PIGEON*?

Pigeon is designed to be a tool for rapidly and systematically visualizing large sets of designs. It is implemented as a Web tool to free the user from managing multiple installations, software dependencies, and upgrades. It is intended to be quickly learnt by users with no prior programming experience, used without downloading software, and to be easily integrated with other software tools. We have integrated *Pigeon* with several tools within the *Clotho*[3] framework and used it for visualizing hundreds of designs in ongoing work with the Voigt Lab (MIT). Many scripting languages allow ways to interact with Web tools like *Pigeon*: we provide a Perl/wget script on the Pigeon Web page to illustrate how this could be done. Its current syntax and vocabulary appear to be sufficient for a number of common visualization needs, according to current user feedback. Six users, including three biologists, and six developers in our lab have learned *Pigeon* syntax and incorporated it into their tools or workflows. *Pigeon* is built with free software tools and interoperable standards, and it can incorporate new commands, support multiple glyph sets and styles, and more complex layout and visualization algorithms easily. Since *Pigeon* is an abstract translator of programs to images, it can include, but is not restricted to, any biological standards, such as SBOL. If standardized, a language like the one used in *Pigeon* specifications could be used for visualizing designs within other software tools, Web pages, or repositories via embedded plugins or Web services. Existing synthetic biology tools like TinkerCell,[4] Spectacles,[3] DeviceEditor,[5] GenoCAD,[6] and others do contain methods for visualizing designs. However, they are not designed specifically for visualization of designs and, with the exception of TinkerCell, do not appear to address all of the visualization goals of *Pigeon* such as support for multiple glyph sets; an easily learnt, parsed, and satisfied syntax; unconstrained programmatic generation of

Table 1. *Pigeon* Programs for Producing the Design Visualizations in Figure 1[a]

(A)	
program ⟨command⟩ ⟨label⟩ ⟨color⟩ [nl]	description of glyph produced
<t t	inverted black T labeled "t"
<c c2 2	dark blue left-pointing pentagon labeled "c2"
<r r1 1	light blue bottom semicircle labeled "r1"
<p p1 6	red inverted right-turn arrow labled "p1"
p p2 2	dark blue right-turn arrow labled "p2"
c c1 6	red right-pointing pentagon labeled "c1"
t t	black T labeled "t"
⟨c label⟩ ind or rep ⟨p label⟩	Arcs drawn
# Arcs	Arc specification start (mandatory marker)
c1 rep p1	from any c1 to any p1, draw a barred arc
c2 rep p2	from any c2 to any p2, draw a barred arc
y rep c1-p1	from label y to any c1-p1 arc, draw a barred arc
x rep c2-p2	from label x to any c2-p2 arc, draw a barred arc
(B)	(C)
p p 13 nl	p p2 2
r r 13 nl	p p6 6
c tetR 13	r r 7 nl
t t 13 nl	c c7 7
p P_LtetO 2	t
>FRT_f 2	p pA
<p pBAD 6	r r 2 nl
r r 1 nl	c c2 2
c flp_e 2	t
d ssrA 12	p pA
t	p pB
<FRT_r 2	r r 5 nl
>loxP_f 6	c c5 5
<p P_A1lacO 4	t
r r 5 nl	p pB
c cre 6	r r 6 nl
d ssrA 12	c c6 6
t	t
<loxP_r 6	p p7 7
r r 3 nl	p p5 5
c gfp 4	r r 4 nl
t	c gfp 4
# Arcs	t
tetR rep P_LtetO	# Arcs
aTc rep tetR-P_LtetO	c2 rep p2
Ara ind pBAD	c5 rep p5
IPTG ind P_A1lacO	c6 rep p6
	c7 rep p7
	A ind pA
	B ind pB

[a] An explanation is provided for the first program.

images; and easy accessibility via the Web. Through its "glyph themes",[7] TinkerCell supports most of the visualization goals but is not Web-accessible.

EXTENDING *PIGEON*

Pigeon was concieved as a tool for quickly rendering designs proposed during design prototyping by scientists and by other software tools. Though *Pigeon* enables the visualization of many common classes of designs, it does not comprehensively allow the expression of all concievable synthetic biology designs. *Pigeon* lacks the commands and glyphs for representing many of the objects and relationships present in the broad synthetic biology lexicon, e.g., designs involving micro-RNA or protein–protein interactions. As we make *Pigeon* available to the wider community, we plan to extend its vocabulary with continual feedback and contributions from the community. We will accept proposals for new commands, glyphs, relationships, and layout through *Pigeon* the Web page and consider incorporating them into new versions of *Pigeon* while preserving executability of user programs written for earlier versions.

AUTHOR INFORMATION

Corresponding Author

*E-mail: swapnilb@bu.edu.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We thank Alec Neilsen, Michael Smanski, Christopher Voigt, Michal Galdzicki, Deepak Chandran, Herbert Sauro, Nathan Hillson, Jackie Quinn, Cesar Rodriguez, members of the CIDAR Lab, and members of the SBOL Visual community for discussions and feedback.

REFERENCES

(1) Quinn, J., Beal, J., Bhatia, S., Cai, P., Chen, J., Clancy, K., Hillson, N., Galdzicki, M., Maheshwari, A., P, U., Pocock, M., Rodriguez, C., Stan, G.-B., and Endy, D. (2013) Synthetic Biology Open Language Visual (SBOL Visual) Version 1.0.0. *BioBricks Found. RFC*, DOI: 1721.1/78249.

(2) Harrower, M. A., and Brewer, C. A. (2003) ColorBrewer.org: An online tool for selecting color schemes for maps. *Cartogr. J. 1*, 27–37.

(3) Xia, B., Bhatia, S., Bubenheim, B., Dadgar, M., Densmore, D., and Anderson, J. C. (2011) Developer's and user's guide to Clotho v2.0 a software platform for the creation of synthetic biological systems. *Methods Enzymol. 498*, 97–135.

(4) Chandran, D., Bergmann, F., and Sauro, H. (2009) TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng. 3*, 19.

(5) Chen, J., Densmore, D., Ham, T. S., Keasling, J. D., and Hillson, N. J. (2012) DeviceEditor visual biological CAD canvas. *J. Biol. Eng. 6*, 1.

(6) Czar, M. J., Cai, Y., and Peccoud, J. (2009) Writing DNA with GenoCAD. *Nucleic Acids Res. 37*, W40–W47.

(7) Chandran, D. (2011) TinkerCell: Computer-Aided Design for Synthetic Biology. Ph.D. Thesis. University of Washington: Seattle, WA, USA.

(8) Gardner, T. S., Cantor, C. R., and Collins, J. J. (2000) Construction of a genetic toggle switch in *Escherichia coli*. *Nature 403*, 339.

(9) Friedland, A. E., Lu, T. K., Wang, X., Shi, D., Church, G., and Collins, J. J. (2009) Synthetic gene networks that count. *Science 324*, 1199–1202.

(10) Tamsir, A., Tabor, J. J., and Voigt, C. A. (2011) Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature 469*, 212–215.

(11) Alternatively, colors may also be specified with names such as lightblue, darkblue, lightgreen, darkgreen, and so on.