A Logic Programming Language for Computational Nucleic Acid Devices

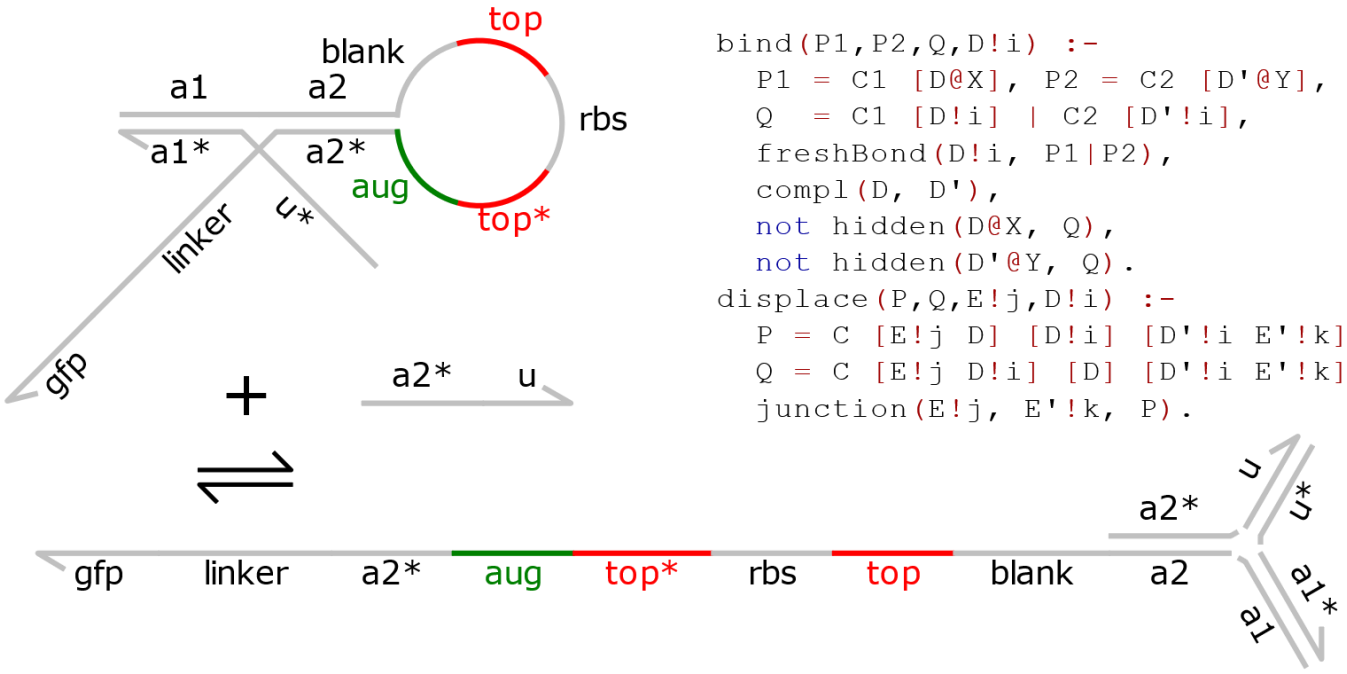
Carlo Spaccasassi[1], Matthew R. Lakin[2] and Andrew Phillips[1]

[1] *Microsoft Research, Cambridge, UK*
[2] *Department of Computer Science, University of New Mexico, Albuquerque, NM, USA*
Email: t-caspac@microsoft.com

Computational nucleic acid devices show great potential for enabling a broad range of biotechnology applications, including smart probes for molecular biology research, in vitro assembly of complex compounds, high-precision in vitro disease diagnosis and, ultimately, computational theranostics inside living cells. This diversity of applications is supported by a range of implementation strategies, including nucleic acid strand displacement, localization to substrates, and the use of enzymes with polymerase, nickase, and exonuclease functionality. However, existing computational design tools are unable to account for these strategies in a unified manner.

We present a logic programming language[3] that allows a broad range of computational nucleic acid systems to be designed and analysed. The language extends standard logic programming with a novel equational theory to express nucleic acid molecular motifs. It automatically identifies matching motifs present in the full system, in order to apply a specified transformation expressed as a logical rule. The language supports the definition of logic predicates, which provide constraints that need to be satisfied in order for a given rule to be applied.

Our language can encode the semantics of nucleic strand displacement systems with complex topologies, previous extensions to the Visual DSD language[2], as well as new extensions including the encoding of kinetic rate hypotheses, together with computation performed by a broad range of enzymes. More importantly, our approach is extensible in that new nucleic acid implementation strategies can be encoded simply by defining new logic predicates. Thus, our approach lays the foundation for a unifying framework for the design of computational nucleic acid devices.




```
bind(P1,P2,Q,D!i) :-
  P1 = C1 [D@X], P2 = C2 [D'@Y],
  Q  = C1 [D!i] | C2 [D'!i],
  freshBond(D!i, P1|P2),
  compl(D, D'),
  not hidden(D@X, Q),
  not hidden(D'@Y, Q).
displace(P,Q,E!j,D!i) :-
  P = C [E!j D] [D!i] [D'!i E'!k]
  Q = C [E!j D!i] [D] [D'!i E'!k]
  junction(E!j, E'!k, P).
```

Figure 1: Part of a chemical reaction network for a ribocomputing AND gate[1], and corresponding logic program code to express domain binding and strand displacement.

References

[1] Green, A.A., Kim, J., Ma, D., Silver, P.A., Collins, J.J., Yin, P.: Complex cellular logic computation using ribocomputing devices. Nature 548(7665), 117 (2017)
[2] Lakin, M.R., Youssef, S., Polo, F., Emmott, S., Phillips, A.: Visual DSD: a design and analysis tool for DNA strand displacement systems. Bioinformatics 27(22), 3211{3213 (2011)
[3] Spaccasassi, C., Lakin, M.R., Phillips, A.: A logic programming language for computational nucleic acid devices. ACS Synthetic Biology (2018). https://doi.org/10.1021/acssynbio.8b00229