Increasing Redundancy Exponentially Reduces Error Rates during Algorithmic Self-Assembly

ACS Nano ASAP, 2015: http://pubs.acs.org/doi/abs/10.1021/nn507493s
Authors: Rebecca Schulman, Christina Wright, and Erik Winfree

Recommended and a commentary by Randall D. Kamien, University of Pennsylvania

When riding your bike or driving your car have you ever zigged when you should have zagged? If you do it how can you possibly expect non-sentient molecules, colloids, and other building blocks to always know where to go? One can tune the shapes (entropic and enthalpic) of the colloids [1], construct complex interaction potentials between them [2], or make a heterogeneous mix of components with specific interactions [3]. One can arrange for a detailed and delicate experimental protocol to help the colloids along at critical moments [4] either in an open-loop or closed-loop situation, the latter requiring the introduction of various "demons" and "daemons" to watch over the assembly [5]. These problems live at the nexus of statistical mechanics in and out of equilibrium, geometry, and information theory. In recent work by Schulman, Wright, and Winfree, the authors have developed, deployed, and experimentally tested another route to faithful production. In a vein akin to kinetic proofreading [6] and error-correcting codes [7], they have created building blocks that have a double-checking feature that halts the growth of materials that miss the targets.

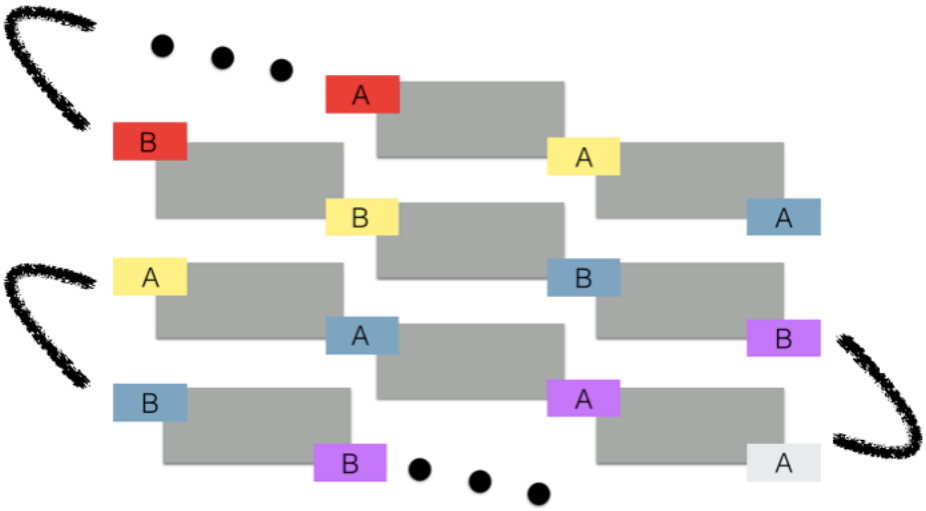

Figure 1. Building the weft (the zig-zag).

They have built on their earlier work of programming DNA tiles. Each tile has four active corners that have unpaired DNA strands in complimentary pairs, for instance upper left to lower right and lower right to upper left. The sequences have been optimized to make the binding as specific as possible. To build a two-dimensional pattern takes two steps; first, a weft is made as shown in Figure 1 (the weft is strand that moves from left to right as one weaves. The warp is perpendicular to the weft.). The colored blocks labeled A and B hold the tiles together along with end caps (not shown) that make the weft into a single zig-zag. Why the colors and labels? Each color and label is a different strand/complimentary-

strand link and the colors are constant along lines going from bottom left to top right. The two types (for instance, "blue A" and "blue B") are necessary so that the doubled end caps have a unique position enforcing the connectivity of the weft – an A row grows to the right and a B row grows to the left.

Now the replication. Along one strand of the weft one chooses a binary code, represented by yellow and green in Figure 2. After programming a row of data the available blocks will automatically propagate the pattern over and over again, forming an arbitrarily long ribbon. All that is required is that there are no "not" tiles that change yellow to green or green to yellow from lower left to upper right.

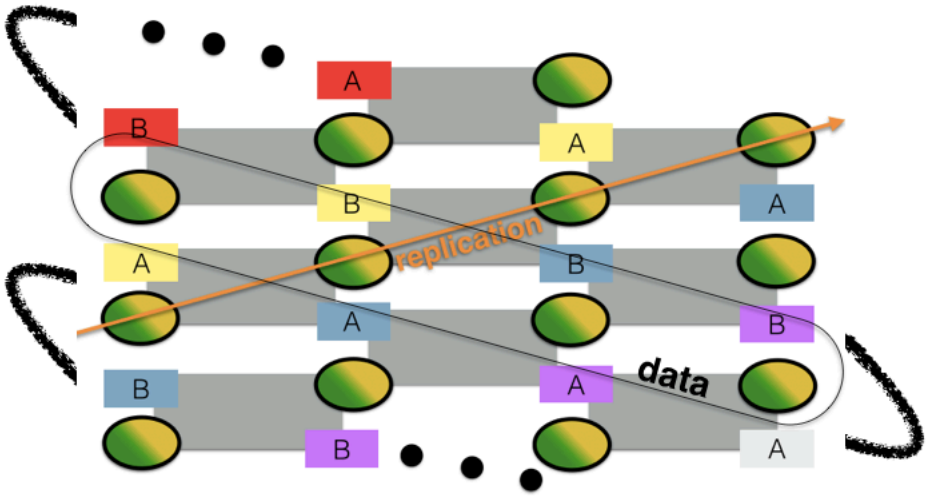

Figure 2. Programming the warp. We can pick a sequence of yellow and green bits along the weft, the data (circled), and have it self propagate as the ribbon grows along the warp.

This modality was pursued by some of the authors in prior work [8]: now they have embellished their assembly procedure by copying each of the bits, for instance, copying each bit so that YGYY becomes YYGGYYYY – twice the bits. That might seem just like an extra layer of complexity that could be achieved by using fatter tiles. But, just like the presence of an intermediate state in kinetic proofreading, doubling the data makes the replication more and more reliable. The left Y and the right Y are made of different blocks. Once a right Y, for instance, attaches to the ribbon, it favors the left Y next. But if, by accident, a right G attaches in the right Y's spot then there is no next tile that fits. The left G is not happy because of the left Y in the preceeding pick (row) and the left Y does not want to be next to a right G. Two mistakes are needed to keep the ribbon growing (see Figure 3 in the paper). Tripling the redundancy only makes this effect stronger. The probability of stopping grows as a power of the redundancy – error correction! This paper systematically explores this qualitative insight in [9] by tripling and quadrupling the message to achieve reliable propagation of information.

Not unlike choosing the right team, the secret is to have intelligent designers who can build reliable building blocks.

[1] J. A. Millan, D. Ortiz, G. van Anders, and S.C. Glotzer, "Self-assembly of Archimedean tilings with enthalpically and entropically patchy polygons", ACS Nano **8** (2014) 2918.
[2] M. Engel, P.F. Damasceno, C.L. Phillips, and S.C. Glotzer, "Computational self-assembly of a one-component icosahedral quasicrystal", Nature materials **14** (2015) 109.
[3] A. Murugan, Z. Zeravcic, M.P. Brenner, and S. Leibler, "Multifarious assembly mixtures: Systems allowing retrieval of diverse stored structures", PNAS **112** (2015) 54.
[4] W.M. Jacobs, A. Reihnardt, and D. Frenkel, "Rational design of self-assembly pathways for complex multicomponent structures", PNAS **112** (2015) 6313.
[5] S. Deffner and C. Jarzynski, "Information Processing and the Second Law of Thermodynamics: An Inclusive, Hamiltonian Approach", PRX **3** (2013) 041003.
[6] J.J. Hopfield, "Kinetic Proofreading: A New Mechanism for Reducing Errors in Biosynthetic Processes Requiring High Specificity", PNAS **71** (1974) 4135; J. Ninio, "Kinetic amplification of enzyme discrimination", Biochimie **57** (1975) 587.
[7] C.E. Shannon, "A Mathematical Theory of Communication", The Bell System Technical Journal **27** (1948) 379.
[8] R. Schulman and E. Winfree, "Synthesis of crystals with a programmable kinetic barrier to nucleation", PNAS **104** (2007) 15236.
[9] R.D. Barish, R. Schulman, P.W.K. Rothemund, and E. Winfree, "An information-bearing seed for nucleating algorithmic self-assembly", PNAS **106** (2009) 6054.