27.2 The Tall Thin Molecular Programmer

Erik Winfree

California Institute of Technology, Pasadena, CA

Solid-state electronic circuits have grown at an astounding rate since the invention of the transistor in 1947, with transistor counts roughly doubling every two years and now exceeding 100 billion on a single chip. One reason this was possible is that computers are an information technology, which permits engineering methods that manage the inherent complexity. Biology establishes that chemistry also is an information technology: all biomolecules are produced according to instructions encoded in DNA, and the resulting biochemical algorithms guide the self-organization of each organism. The potential design space of chemical systems includes all of biology as well as other forms of information-based chemistry that have yet to be explored. Will the engineering of information-based chemical systems follow in the footsteps of information-based electronic systems, with the design complexity increasing exponentially for decades? What lessons do the early years of the computer revolution offer to this field, especially with respect to how systematic design can help master often bewildering complexity? Now having accumulated a 40-year history, DNA nanotechnology provides a case study for these questions.

DNA nanotechnology began as a theoretical vision for how DNA molecules could be designed that would robustly self-assemble into a crystal with precisely controlled structure and geometry [1]. Along the way to achieving that goal, it was realized that sequence-directed molecular self-assembly enables a remarkable range of nanoscale devices and systems to be systematically designed, including arbitrarily shaped finite structures that self-assemble in solution, scaffolds for controlled geometric arrangement of molecules such as proteins and carbon nanotubes, mechanical hinges and motors and springs, molecule-sensing triggers and logic gates, information processing circuits, and robotic systems integrating multiple subsystems and modalities [2,3]. Figure 1 gives examples of self-assembled structures built over the past 40 years, as well as molecules and systems developed during the past 30 years of DNA computing.

The fundamental design basis for these systems is the simplicity of Watson-Crick base pairing in DNA and the associated biophysical mechanisms that govern self-assembly and conformational changes of the molecules. For example, as shown in the bottom left of Figure 1, if a single-stranded DNA molecule with the directed sequence 'ATTACCATGAGCTG' is in solution with 'GTACCTGCAGCTCA', the complementary subsequences will bind to each other when they meet, forming a double-helical region 'TGAGCTG/CAGCTCA' that holds the molecules together in an anti-parallel orientation, 'T' with 'A', 'G' with 'C', 'A' with 'T', 'C' with 'G', etc. Thus, the subsequences serve as information-bearing 'barcodes' that direct self-assembly. Complementary subsequence within an individual strand (for example, one named 'd' and its complement called '$\overline{d}$') can direct folding of the strand itself. Designing a set of DNA molecules that self-assemble into a target shape boils down to designing sequences that are sufficiently orthogonal to ensure specific interactions. This was the essential insight of Nadrian Seeman 40 years ago [1]. More sophisticated molecular machines incorporating structural reconfiguration – moving parts – have been subsequently developed using the related dynamical processes of branch migration and strand displacement [2,3]. These processes work naturally in solutions containing no more than water and salt – that is, no biological enzymes are needed – and the designed sequences can be synthesized as single-stranded DNA molecules for costs of pennies per nucleotide.

How can we measure the increase of complexity in DNA nanotechnology and DNA computing? We restrict our attention to systems designed exclusively using DNA, which we may think of as the information-processing core for programmable molecular systems. Figure 2 plots the complexity of a selection of landmark publications in DNA nanotechnology, classified into three groups. (1) The mass of well-defined self-assembled structures that were produced in a multi-stage process (e.g. first make part A, then part B, then mix them). This measures our ability to manufacture molecularly precise objects, by any means available. (2) The sequence design complexity (i.e. the total length of synthesized DNA strands) for well-defined structures that self-assemble in a single "one-pot" reaction. This measures our control over how DNA sequence information autonomously directs the fabrication. (3) The sequence design complexity of systems whose function is to compute, which imposes more severe design challenges on molecular function beyond just structural integrity.

In all three cases, the complexity appears to be increasing exponentially, with doubling times between every two years to every four years. If these trends continue, what can we predict for the coming 40 years of DNA nanotechnology research? One reference point is the size of a typical bacterium, such as E. coli, which is about 1 cubic micron. A nucleotide is about 1 cubic nanometer, so to self-assemble a life-size molecular "statue" of a bacterium would require 10^9 nucleotides – which we can expect to be feasible within 15 years. If we want a simple procedure – such as just sending an email to a DNA synthesis company, mixing the strands in a single test tube at the right temperature, and waiting for billions of bacterium statues to self-assemble by themselves – then we may have to wait 30 years. If we want the molecular analog of a cell-scale cuckoo clock chock-full of moving parts and programmable autonomous logic circuitry (which we might call a "molecular robot") then we may have to wait 50 years.

What bottlenecks might prevent future exponential growth of DNA nanotechnology, and how might we avoid them? Examining the history of the electronics industry may suggest common principles that could be in play despite the huge differences in technology. Exponential growth of solid-state circuit technology was not a straightforward process; it required a series of fundamental innovations and breakthroughs at different times during its development. What point on this growth curve is most analogous to the current state of molecular information technology? Self-assembled DNA structures, 40 years from the design of the first four-way junction, have increased in complexity 10^5-fold; microprocessor chips reached 10^5 transistors around 1982, 35 years after the invention of a single transistor.

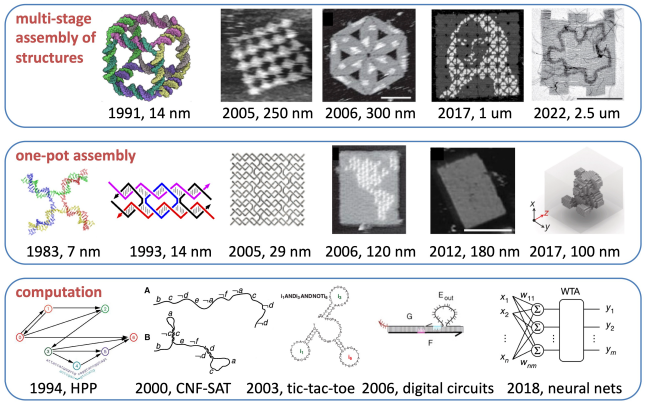
This was about the time of the Mead-Conway VLSI chip design revolution – their book [4] came out in 1980, walking students through transistor device physics and lithography all the way to circuit and microprocessor architecture design. The book presented an integrated view of principles used for managing the complexities of chip design, including the digital abstraction that enables decoupling of logical design from analog physics, the design rules for fabrication that facilitate consistently reliable circuit layout, and the abstraction hierarchy that guides designs from high-level programming languages down to the physical implementation. Remarkably, these and other design principles simplified the process enough that a single person – perhaps a student – could understand the entire flow well enough to do the design from top to bottom using a structured approach. Carver Mead called that person the "Tall Thin Computer Engineer".

DNA nanotechnology provides an almost unique corner of chemical engineering where analogous principles can be put to the test. The past decade has seen efforts to design biochemical circuits using rigorous application of the digital abstraction (Figure 3) or its generalization to higher-dimensional discrete restoration by attractor networks (Figure 4), to explore the consequences of working strictly within design rules that enable scalable systematic methods for managing complexity (Figure 5), and to demonstrate how layered abstraction hierarchies can support the design of complex circuits and self-assembling structures (Figure 6). Computer-aided design software for self-assembled DNA structures [5] and programming languages that compile down to DNA strand displacement circuits [6] are now de facto in some branches of DNA nanotechnology research. These types of efforts, treating molecular engineering as an information technology where lessons from computer engineering ought to apply, is now often called "molecular programming".

In another parallel to the 1980's, many of the most successful recent advances in molecular programming have come from individual researchers who personally encompass the full stack, from characterizing critical molecular biophysics to writing software that compiles high-level designs into DNA sequences. Following Carver Mead, we might call such people "Tall Thin Molecular Programmers". Their role in the development of DNA nanotechnology is particularly critical right now, as the abstractions and methods currently used to handle the complexity of molecular systems today should not be considered optimal or even workable for continued scaling – finding new effective abstractions and design strategies up and down the stack will require an integrated view of how the different layers interact with each other.

[1] N. C. Seeman, "Nucleic acid junctions and lattices", Journal of Theoretical Biology, vol 99, pp 237-247, 1982.
[2] H. Ramezani and H. Dietz, "Building machines with DNA molecules", Nature Reviews Genetics, vol 21, pp 5-26, 2020.
[3] F. C. Simmel et al, "Principles and applications of nucleic acid strand displacement reactions", Chemical Reviews, vol 119, pp 6326-6369, 2019.
[4] C. Mead and L. Conway, "Introduction to VLSI Systems", Addison-Wesley, 1980.
[5] M. Glaser et al, "The art of designing DNA nanostructures with CAD software", Molecules, vol 26, 2287, 2021.
[6] M. R. Lakin and A. Phillips, "Domain-specific programming languages for computational nucleic acid systems", ACS Synthetic Biology, vol 9, pp 1499-1513, 2020.

DNA nanotechnology and DNA computing



Seeman, Liu, Yan, Rothemund, Qian, Shih, Yin, Adleman, Hagiya, Stojanovic, Seelig, Qian, et al

Figure 27.2.1: Historical examples of DNA nanotechnology and DNA computing constructs.

Growth of DNA nanotechnology

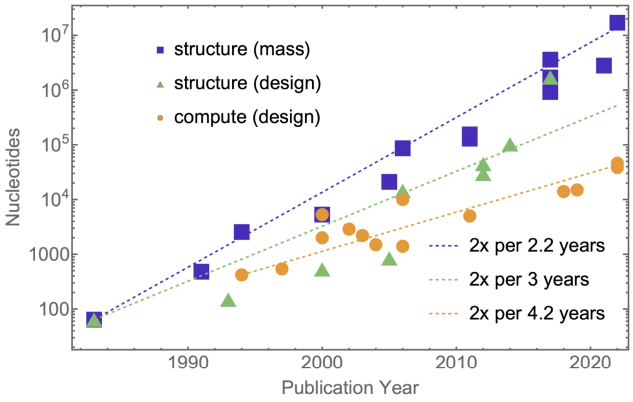
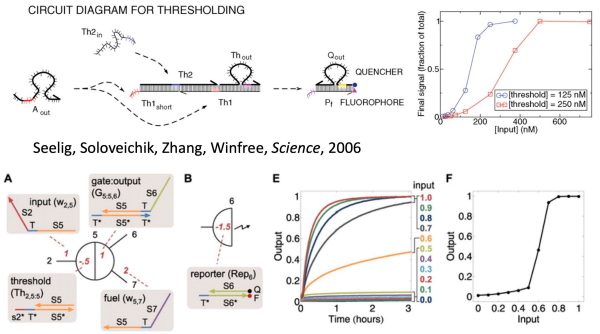


Figure 27.2.2: Scaling of complexity for objects and designs.

Analog physics and digital abstraction

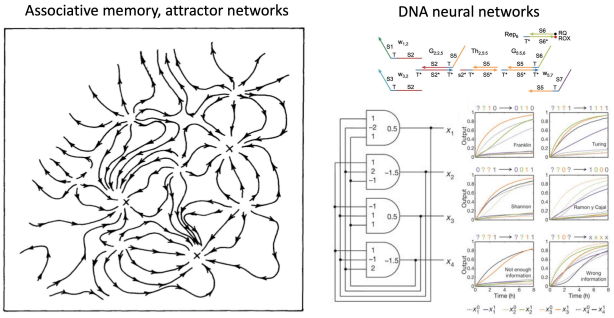


Seelig, Soloveichik, Zhang, Winfree, *Science*, 2006

Qian, Winfree, *Science*, 2011

Figure 27.2.3: DNA strand displacement circuits that employed the digital abstraction.

Analog physics and digital abstraction



Hopfield, *Complexity in Information Theory*, 1988

Qian, Winfree, Bruck, *Nature*, 2011

Figure 27.2.4: Neural-network-like restoration for more robust non-binary digital abstractions.

Design rules for fabrication

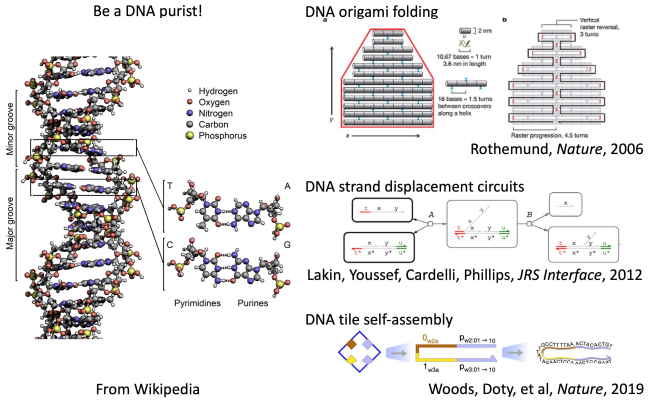
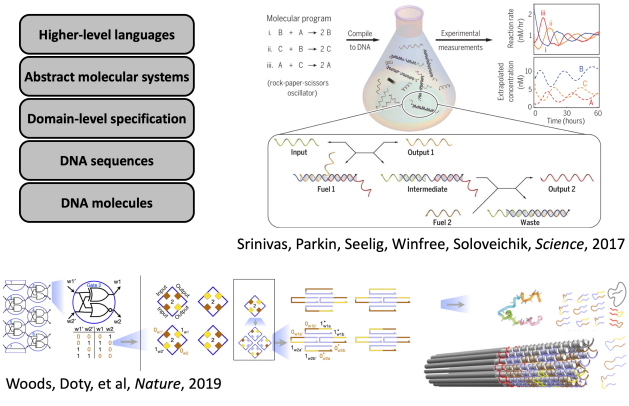


Figure 27.2.5: Design rules in DNA nanotechnology limit chemical options but enable effective abstractions.

The abstraction hierarchy



Srinivas, Parkin, Seelig, Winfree, Soloveichik, *Science*, 2017

Woods, Doty, et al, *Nature*, 2019

Figure 27.2.6: Abstraction hierarchies are being prototyped in molecular system design.