Computing by molecular self-assembly

Nataša Jonoska1,* and Nadrian C. Seeman2

1Department of Mathematics and Statistics, University of South Florida, Tampa, FL 33620, USA
2Department of Chemistry, New York University, New York, NY 10003, USA

The paper reviews two computing models by DNA self-assembly whose proof of principal have recently been experimentally confirmed. The first model incorporates DNA nano-devices and triple crossover DNA molecules to algorithmically arrange non-DNA species. This is achieved by simulating a finite-state automaton with output where golden nanoparticles are assembled to read-out the result. In the second model, a complex DNA molecule representing a graph emerges as a solution of a computational problem. This supports the idea that in molecular self-assembly computing, it may be necessary to develop the notion of shape processing besides the classical approach through symbol processing.

Keywords: DNA self-assembly; biomolecular computing; nano-devices

1. ALGORITHMIC SELF-ASSEMBLY

In the last few decades, research performed in biology, chemistry and physics has resulted in an explosion of new findings about molecular interactions. These findings often reveal transfer of information at a molecular level resulting in proliferation of the science of computing within established, on a first glance unrelated, scientific fields. The notion of ‘computing’—up until recently a theoretical concept—acquires a new meaning within these intrinsically experimental disciplines. The perception of ‘computation’ has evolved to mean an environment modification through repetitive applications of a finite set of simple local rules. At a molecular scale, these local interactions are often covalent, hydrogen or ionic bonds. Computing models based on molecular interactions have been developed theoretically since Head’s [1] introduction of splicing systems in the mid 1980s, and experimentally since Adleman’s [2] first experiment in 1994 solving a computational problem solely by DNA molecules. Today, we have annual scientific conferences on biomolecular computing [3] and unconventional computing [4], where many new models are being introduced and developed.

Molecular self-assembly appears as the central principle in many models of biomolecular computing. Self-assembly is currently an imprecisely defined notion understood to be a process in which individual elements self-organize in a larger more complex structure. This self-organization ranges from cosmic levels (such as crystal formations or protein folding). Much of designed biomolecular self-assembly is based on DNA self-assembly, although other examples of self-assembly—such as those involving proteins—exist in nature. The inherently informational character and its predictable double-helical geometry makes DNA an attractive molecule in applications that entail targeted assembly. DNA-based bottom-up assembly, pioneered roughly 30 years ago [5], has been explored by at least 60 laboratories worldwide in recent years. Within the past 15 years, much of these explorations have been in search of understanding molecular information processing through algorithmic, programmable self-assembled structures.

Synthetic DNA molecules have been designed and shown to assemble into branched species [6,7], and more complex, structurally robust, species that entail the lateral fusion of DNA double helices [8], such as DNA double crossover (DX) molecules [9], triple crossover (TX) molecules [10] or panemic crossover (PX) molecules. DX and TX molecules have been used as tiles and building blocks for large nanoscale arrays [11,12]. Winfree [11] introduced the tile assembly model and showed that two-dimensional self-assembled arrays made of DX or TX DNA tiles can simulate the dynamics of a bounded one-dimensional cellular automaton and so are capable of potentially performing computations as a universal Turing machine. Several successful experiments have confirmed computation by array-like DNA self-assembly, such as the binary addition (simulation of XOR) using TX molecules (tiles) [13], Sierpinski triangle assembly [14,15] and a binary counter [16] by DX molecules.

In addition to the two-dimensional arrays, three-dimensional structures such as a cube [17], a truncated octahedron [18], tetrahedron [19] and arbitrary graphs
have been constructed from DNA duplex and junction molecules. In addition to structural targets, a variety of topological figures have been constructed using the methods of structural DNA nanotechnology [23]. The construction of approximately 100 nm DNA nanostructures was significantly facilitated by Rothemund’s [24] DNA origami where a standard single-stranded vector plasmid is used to outline a shape, whereas short DNA strands connect portions of the plasmid, fixing its shape in a rigid form. Since its appearance, DNA origami has been used as a seed for arranging DX tiles in an array [16], nanotube constructions [25] and most recently as templates for large DNA based tiles arranged in a two-dimensional crystalline array [26].

‘DNA fuel’ strands introduced by Yurke [27] provided devices whose activity is controlled isothermally by DNA strands and hence were sequence-dependent [27,28]. Also based on strand displacement, structures that can perform simple ‘walking’ on an arranged platform have been reported [29–32]. The strand displacement techniques were recently used for a large number of enzyme-free logic gates [33,34] and also for simulating a neural network type of computation [35].

This review describes two realizations of DNA self-assembly computation. Both are recently reported experiments [22,36] depicting two distinct approaches. Section 2 summarizes the results reported by Chakraborty et al. [36] on molecular simulation of a finite-state automaton that divides by 3. This molecular computing model incorporated strand displacement devices to algorithmically arrange TX tiles, and moreover, the result of the computation was an algorithmic arrangement of non-DNA (gold) nanoparticles. Although arrangements of gold nanoparticles in DNA-scaffolded periodic arrays have been reported earlier [37–39], this is, to our knowledge, the first algorithmic arrangement of non-DNA particles as a result of computation by self-assembly.

Section 3 describes an experiment where an assembly of a complex DNA nano-object representing a graph structure embedded in space represents a solution of a non-trivial graph theoretic problem [22]. The three-colourability of a graph with six vertices was demonstrated by assembling a DNA molecule with connectivity of the graph structure itself. Moreover, the general method used in the construction distinguishes by: (i) requiring only four laboratory steps for a solution of an NP-complete problem regardless of the size of the problem, and (ii) a graph-like structure survives an enzymatic system if and only if the graph is three-colourable. These results imply that natural processes and environment in which three-dimensional molecular structures with various functionality can emerge perform ‘structural’ computation. Hence, information processing and computation through molecular self-assembly is a venue where ‘shape processing’ is a viable alternative to classical ‘symbol-based’ computing.

2. DNA SELF-ASSEMBLY OF AN AUTOMATON

This section summarizes the experimental results given by Chakraborty et al. [36]. A more general discussion about transducer-generated arrays and potential arrangements of robotic arms within an array capable of simultaneous movements is described in Dolzhenko et al. [40].

2.1. Formal description

A transducer, also known as a finite-state automaton with output, is a formal system consisting of an input/output alphabet, a transition function, one designated initial state and a finite set of states, some of which are designated terminal. More formally, a finite-state automaton with output or a transducer is a five-tuple \( \tau = (\Sigma, Q, \delta, q_0, F) \), where \( \Sigma \) is a finite alphabet, \( Q \) is a finite set of states, \( q_0 \) is an initial state \( (q_0 \in Q) \), \( F (F \subseteq Q) \) is a set of final (terminal) states and \( \delta \subseteq Q \times \Sigma \times Q \) is the set of transitions. From a given state and an input symbol, the transition function determines the next state, and an output symbol.

We associate a directed labelled graph to a transducer: the set of vertices is the set of states \( Q \) and the directed edges are transitions in \( \delta \), such that an edge \( e = (q, a, a', q') \) starts at \( q \) and terminates at \( q' \), has an input label \( a \) and an output label \( a' \). The input and the output labels of the edges are naturally extended to input and output labels of paths in the transducer. We say that a word \( w \) is accepted by a transducer \( \tau = (\Sigma, Q, \delta, q_0, F) \) if there is a path \( p = e_1 e_2 . . . e_n \) with input label \( w \) which starts at \( q_0 \) and terminates with a state in \( F \). The path \( p \) in this case is called an accepting path for \( w \). The output label of \( p \) is said to be an output of \( \tau \).

Transducers can be simulated by assembly of Wang tiles [41,42]. A finite set of distinct unit squares with coloured edges is called a set of Wang prototiles. Each prototile has an arbitrarily large number of copies called tiles. A tile \( \xi \) with left edge coloured \( l \), bottom edge coloured \( b \), top edge coloured \( t \) and right edge coloured \( r \) is denoted with \( \xi = [l, b, t, r] \). No rotation or reflexion of the tiles is allowed. Two tiles \( \xi = [l, b, t, r] \) and \( \xi' = [l', b', t', r'] \) can be placed next to each other, \( \xi \) to the left of \( \xi' \) if and only if \( r = t' \) and \( \xi' \) on top of \( \xi \) if and only if \( t = b' \).

To each transducer \( \tau \) we associate a collection \( T_\tau \) of Wang tiles consisting of input, output and computational tiles. Given a transducer \( \tau \), to each transition of the form \( (q, a, a', q') \), we associate a prototile \( [q, a, a', q'] \) in \( T_\tau \) as presented in figure 1. For each symbol \( a \in \Sigma \), the collection \( T_\tau \) contains an input prototile \( \xi_a = [c, \beta_1, a, c] \) and an output prototile \( \xi_a = [c', a, \beta_1, c'] \). The colours \( \beta_1, \beta_2, \beta_3, \beta_4 \), called borders, represent one of the left, bottom, right and top border, respectively, and \( c, c' \) are connect colours used to connect input and output tiles. Additional auxiliary tiles indicating start of the input, end of the input, start...
of computation and end of computation are also included in $T$, but are not an essential part of the computation.

With these sets of input and output prototiles, every computation of $t$ can be modelled as a tiled rectangle surrounded by boundary colours. In particular, one run of the transducer $t$ reading an input of length $n$ and producing an output is obtained with a tiled $n/C^2$ rectangle such that the sides of the rectangle are coloured with boundary colours. As an example, consider the transducer $t$ presented in figure 2a. It has initial and terminal state $s_0$ with input/output alphabet $\{0, 1\}$. It accepts the set of binary strings that represent numbers divisible by 3. The states $s_0, s_1, s_2$ represent the possible remainder (0, 1, 2, respectively) of the division of the input string with 3. The output is the result of the division of the binary string with 3. On input 10101 (21 in decimal), the transducer gives the output 00111 (7 in decimal). The 7/C^2 rectangle of assembled Wang tiles simulating this division is illustrated in figure 2b. The bottom row of tiles consists of the input tiles, the middle row corresponds to the computation by the transducer and the top row consists of the output tiles. It is not difficult to see that the tiles can be modified to simulate composition and iteration of transducers which is known to have computational power of the universal Turing machine [42].

### 2.2. Molecular implementation

The input of the transducer is established by a sequence of robust two-states [28] DNA devices attached to a six domain-flat (6DF) motif tiles encoding input symbols at their ends. The sequence-dependent two-state nanomechanical device called PX-JX2 whose machine cycle is shown in figure 3a, is guided by the addition of set strands to the solution that forms its environment. The two states of the device differ by rotation of a half-turn established by the green and magenta set strands. The set strands have short, unpaired extensions on them such that the state of the device is changed by binding the full biotin-tailed complements of green or magenta strands, removing them from the solution by magnetic streptavidin beads, and then adding the other strands. The sequence-driven nature of the device means that many different devices can be constructed, each of which is individually addressable.
The computational Wang tiles representing transducer positions are simulated with TX DNA molecules. It has been shown that such TX DNA molecules can self-assemble in an array [10], and they have been molecular copies as separately annealed DNA TX molecules. No rotation of the tiles is assured by the rotational character of the device allows all possible binary strings can be represented as an input. The computational TX molecules are placed on top of the input row and the chelator tiles with gold nanoparticles recognizing the output symbols are attached to the TX tiles. Nanoparticles of 5 nm ‘recognize’ 0 and of 10 nm ‘recognize’ 1.

Thus, the output can be read in a transmission electron microscopy [36]. This layer of chelator tiles corresponds to the top row of the 7 x 3 assembled rectangle in figure 2b.

A schematic of the computational set-up corresponding to the Wang tiles in figure 2b is illustrated schematically in figure 5. In figure 5, a base row of PX-JX2 devices setting up the input of 21 binary is depicted (as in the example shown in figure 2); the middle row of computational tiles is represented by DNA TX molecules encoding the transducer transitions; and the top row of Wang tiles containing the output of the computation is obtained by the sequence of chelator DNA DX motifs with attached gold nanoparticles. Different input sequences are obtained by setting the PX-JX2 devices in different states. Iteration and composition of transducers can be obtained if instead of the chelator tiles, we apply another layer of TX computational tiles which computes on the obtained output as a new input. The gaps between the top duplexes of two consecutive TX tiles fit precisely the bottom duplex of any TX tile. After several layers of TX tiles are applied, the chelator DNA DX tiles with the attached gold nanoparticles can be added.

3. Computing by self-assembling shapes

This section summarizes results of Wu et al. [22]. As mentioned in §1, the review reports assembly of a
complex DNA molecule whose structure encodes a solution to a computational problem. The problem addressed by this experiment is a well-known NP-complete problem, the three-colourability of a graph. A DNA graph structure is self-assembled by junction building blocks if and only if the given instance of the DNA graph structure is self-assembled by junction building blocks if and only if the given instance of the problem has a solution.

Let $G$ be a graph with vertices $V$ and edges $E$. The graph $G$ is three-colourable if there is a function $f : V \rightarrow \{a, b, c\}$ such that whenever two vertices $v, w \in V$ are adjacent (connected by an edge), $f(v) \neq f(w)$. The function $f$ is a colouring of the vertices of $G$ by three colours. So, we say that $G$ is three-colourable if it is possible to colour each vertex such that no two adjacent vertices are coloured with the same colour. The $n$-colourability is defined similarly.

Figure 6a shows an example of a graph with a possible three-colouring. The illustrated solution was obtained experimentally by construction of a DNA molecule whose helical axes conform to the shape of the graph.

A general procedure for solving this and other graph-theoretic problems is described in Jonoska et al. [20], and the theoretical computing model based on this method with introduction of certain complexity classes is described in Jonoska & McColm [45]. In this approach, each vertex of degree $k$ in a given graph $G$ is represented with a $k$-armed branched junction molecule. In the case of $k = 1$, the vertex is represented by a DNA hairpin. The $5\'\prime$-ends of each arm of every $k$-armed junction molecule end with single-stranded extensions. These extensions consist of three parts (labelled $a$, $b$, $c$) and are used to uniquely identify each edge that connects two adjacent vertices. The first and the third part, parts $a$ and $c$ are specific encodings for the edge that is represented by the given arm of the molecule, whereas the middle portion, the part $b$, encodes the colour of the vertex. The middle part of the encoding is the same for all arms of the vertex molecule fixing the colour of the vertex. For each vertex, three types of molecules are needed, each corresponding to one of the three possible colours of the vertex. They are identical except for the middle portion of the single-stranded extensions of the arms identifying the colour of the vertex.

For the example presented in figure 6a, one two-, four- and one four-armed branched junction molecule are needed to identify the graph (figure 6b). If vertices $v$ and $w$ are connected by an edge $e$, the $a$ and $c$ DNA sequence of the arm of the $v$-building block representing $e$ is complementary to the corresponding $a$ and $c$ DNA sequence of the $w$-building block arm representing $e$. This is illustrated in the arms of the three- and four-arm molecules corresponding to the edge connecting vertex four with vertex six in figure 6b. Therefore, when the two building blocks are in a solution, parts of the $v$-building block will anneal with the complementary parts of the $w$-building block. In addition, the middle portions $b$ in $v$- and $w$-building blocks are also complementary, they would anneal and would indicate that the two vertices $v$ and $w$ are coloured with the same colour. Each colour sequence contains a cleavage site for a restriction enzyme, such that when two arms of identically coloured vertex building blocks are hybridized the restriction site is exposed and can be
cut by an appropriate enzyme. But if the vertex building blocks encode distinctly coloured vertices, then the mismatched restriction sites form a ‘bubble’ that cannot be cut by a restriction enzyme (figure 6c). One restriction enzyme is used for each colour, hence for the three-colourability, three enzymes are needed. In the experiment reported by Wu et al. [22], the enzymes used to cleave the monochromatic edges were HindIII representing red, BamHI representing blue and EcoRI representing green.

Given the vertex building blocks of a graph, the solution of the three-colourability problem then needs the following four steps, regardless of the size of the graph:

- **anneal**: combine all vertex building blocks molecules in a single test tube and allow the complementary ends to hybridize,
- **ligate**: join the open nicks with a ligase,
- **cleave**: apply three restriction enzymes to destroy the molecular structures containing identically coloured vertices joined by an edge, and
- **extract**: determine whether any DNA graph structure of the size of the input remains; the graph is three-colourable if the structure is detected, otherwise it is not.

Figure 7a shows a possible formation of a DNA graph structure. The vertex colours are indicated and if all nicks are sealed at the location of the arrowheads, there is a single-stranded circular molecule that traverses the whole graph structure. Such a molecule forms a knot in three-dimensional space. Preliminary experiments detecting the graph structure identified two topoisomers [20], proving that the graph structure may conform to two non-identical knots. In the computational experiment [22], only some of the nicks were ligated such that a linear single-stranded molecule traversing all edges at least once, a reporter strand, was formed and the graph structure was easily detectable. The reporter strand is illustrated in figure 7b indicated in purple. Although it is not possible for every graph to be represented as a single circular molecule [46], it has been shown that the vertex building blocks can be designed such that a reporter strand traversing every edge of a given graph at least once exists for every graph [47].

The current techniques limit the above-described method to smaller size graphs (graphs with about 13 vertices and 28 edges [22]), but this experimental solution of the three-colourability shows that, in principle, a complex molecular shape, including mismatched pairs, results from a computation and this complex molecule represents an answer of the computation. Hence, the complex molecular structures, such as those of proteins, resulting from natural self-assembly that undergo enzymatic selection processes can be seen as products of computations.

4. CONCLUDING REMARKS

There are many models for computing by DNA self-assembly that are not covered in this review. Some of them are quite stimulating, such as the autonomous simulation of finite-state automata by Benenson [48] where a finite-state automaton is simulated by applying a restriction enzyme and a ligase, or Stojanovic’s DNA-zyme-based automaton that plays tic–tac–toe [49], or the algorithmic tile assembly model introduced by Winfree [11] where DNA tiles simulate the dynamics of a cellular automaton. Some other self assembly models such as Rothemund’s DNA origami [24] turned out to advance designs in DNA nanotechnology enabling new shapes to be reported on a daily basis. Yurke’s [27] ‘fuel strands’ and isothermal strand displacement-based devices are now taken to a new level by constructing a large number of logic gates that can potentially operate simultaneously [34], and to small walkers that can carry a molecular cargo [29]. Each one of these aspects of algorithmic DNA self-assembly has the potential to develop and possibly drastically influence new material designs, as well as to provide platforms for further studies of molecular interactions. Unfortunately, at the moment, there is no unified theoretical approach that describes all of the different self-assembly models. This task remains as a difficult and challenging problem.

We acknowledge support by grant no. CCF-1117254 from the NSF to both N.J. and N.C.S., grant no. DMS-0900671 to N.J., and the following grants to N.C.S.: GM-29554 from NIGMS, grant no. CTS-0608889 from the NSF, grant no. W911FF-08-C-0057 from ARO, via Pegasus Corporation, MURI W911NF-07-1-0439 from ARO, grant nos.
N000140910181 and N000140911118 from ONR and a grant from the W.M. Keck Foundation.

REFERENCES


3 International Conference on DNA-based computing and molecular programming (annual meetings). See http://www.dna-computing.org/.


1996 Organization of ‘nanocrystal molecules’ using DNA. Nature 362, 609–611. (doi:10.1038/382609a0)


