

RANDOM PHENOMENA IN ALGORITHMIC SELF ASSEMBLY

E. G. Coffman, Jr.
Columbia University

ABSTRACT

Speed of computation and power consumption are two principal parameters of conventional computing devices implemented in microelectronic circuits. As performance of such devices approaches physical limits, new computing paradigms are emerging. This paper focuses on computing by molecular self assembly processes, where computing elements are fashioned from DNA. For purposes of analysis, DNA-based self assembly can be abstracted to growth models in two dimensions where computational elements modeled as tiles are self-assembled one by one, subject to some simple hierarchical rules, to fill a given template encoding a Boolean formula. While molecular computational devices are known to be extremely energy efficient, little is known concerning the fundamental question of computation times. In particular, given a function, we study the time required to determine its value for a given input. Error tolerance is a necessary concomitant issue of any DNA-based computing paradigm. We propose an effective pulsing technique for error recovery and extend our estimates of computing times to this more general model. Our over-all analytical approach establishes a reference theory for stochastic modeling of molecular computing. The mathematics comes down to customized techniques drawn from a surprisingly large number of disparate areas, including probability theory, dynamical systems, optimal control theory, and theory of networks.

1 Introduction

Self assembly is a fundamental process of numerous computational and fabrication technologies in nanoscience. Its efficiency and robustness is critical to the success of the underlying concepts. The research reported here addresses self-assembly systems from the traditional framework of the analysis of algorithms and in so doing establishes a reference theory for self assembly processes. We concentrate on the *performance*, broadly understood, of self-assembly systems within existing computing paradigms.

Our motives are two-fold: for one, the *tiling models* of self-assembly and molecular-scale computation have emerged as a leading paradigm; for the other, there is a continuing need of original research in the *stochastic analysis* of self-assembly systems which takes into account the intrinsic random phenomena and noisiness of computations. Thus, the introduction of techniques central to the *average-case* analysis of algorithms is critical to the development of the field. Providing a reference theory and customized analytical framework facilitates future advances in existing and proposed approaches to the design of molecular self-assembly systems.

To this end we deploy a broad variety of tools from probability theory, dynamical systems, optimal control theory, theory of networks, and simulations in a study of the performance characteristics of self-assembly structures. For example, the variety calls into play ramifications of packing problems (e.g. Random Sequential Absorption in the physics and chemistry literature), particle processes, such as the TASEP process, systems of first-order nonlinear ODEs of chemical kinetics, and properties of random integer sequences.

More specifically, the problems attacked in this paper involve fundamental questions like: “How long does a given structure take to self-assemble (equivalently, how long does it take to perform a DNA-based computation)?,” and “What are the trade-offs between the reliability (error tolerance) and speed of self assembly?”. In the next section, we describe and analyze a reference theory which adds a stochastic component to the Wang tile-models [14] of computation developed by Winfree [16], and which gives insights into the first of the above two questions. In the final section, a checkpointing technique for error tolerance adapted from classical computer operating systems is presented and analyzed as a response to the second of the above two questions.

Other abstractions of molecular self-assembly that apply to polymerization kinetics have also been studied. We content ourselves here with a very brief introduction to this work.

The performance of linear tile self-assembly was studied by Adleman et al [3]. In their model, there are n tile types arranged on a doubly infinite line. These tiles bond to their neighbors with given probabilities, σ_{ij} being the probability that a tile of type i bonds to a tile of type j . An existing bond between tiles of types i and j is broken with probability τ_{ij} , where $\tau_{ij} > 0$ for reversible systems. The attachment of contiguous tiles forms a

supertile. The formation of supertiles can be thought of as manipulations within a language of strings, in this case a language closely related to the class of regular sets.

Adleman et al studied the problem of *n-linear polymerization*, where the system consists of equal quantities of tiles T_1, \dots, T_n , with $n \geq 2$. Only tiles of type i can bond with tiles of type $i + 1$. The tiles are initially placed randomly on the integers and are allowed to bond with their neighbors according to the probabilities σ_{ij} . Once all possible supertiles are formed, the tiles in the system are said to be “tossed” in such a way that when they return to the integer lattice, their sequence is uniformly randomized; i.e., the supertiles and tiles are placed uniformly at random along the integers. The tiles will again be able to join with neighboring tiles or supertiles with probabilities σ_{ij} , and supertiles are allowed to be broken with probabilities τ_{ij} . Adleman et al showed that, if this process is continued, then *n-polymerization* converges to an equilibrium. They also give estimates of convergence rates. A shortcoming of the above model is that a linear self-assembly system can not realize the Turing computability achievable in a two dimensional self-assembly system, such as the tile model of this paper.

The essentially dimensionless self assembly process in [4, 5] is couched in terms of polymerization. It starts with a homogeneous set consisting only of monomers. As the process evolves, monomers combine with each other and with other self-assembled polymers to become larger polymers with the corresponding bonding events determined by given reaction-rate parameters. The process stops when there are no two polymers in the system which can combine (corresponding reaction rates have the value 0). The goal is to determine the *yield* of the process. Based on the bonding rules of the self-assembly process, the yield is defined as the concentration of a pre-specified stable polymer which cannot decompose or combine with other polymers.

Chemical kinetics, in particular reaction rate equations and the chemical networks of Feinberg [8, 7], provide a useful hydrodynamic model of this molecular self-assembly process. The dynamical system of self-assembly becomes a system of first-order, usually nonlinear, ordinary differential equations (ODEs) giving the rates of change of concentrations in terms of the concentrations themselves and the reaction rates. In chemical kinetics, the reaction rates are determined by properties such as the temperature of the

system and the bonding energy between colliding particles. Explicit yield results for special cases can be found in [5] along with experimental characterizations under general assumptions. An interesting phase transition property is discovered, revealing regions of the parameter space where yields are 100%.

2 DNA-based computing times

2.1 Introduction

The elementary logic unit of DNA computing is the *tile* modeled as shown in Figure 1 as a marked or labeled square. In the simplest version, the label values are 0 or 1, and they break down into two input labels on one edge and two output labels on the opposite edge. A computational step consists of one tile bonding to others according to given rules that match input labels of one tile to the output labels of one or two adjacent tiles. Figure 1 shows a simple example. Successive bonding of tiles performs a computation, e.g., the evaluation of a Boolean formula, in a self assembly process guided by a *template* incorporating given inputs. The tiles are DNA-based molecular structures moving randomly, in solution, and capable of functioning independently and in parallel in the self assembly process; the template is needed to properly structure the self assembly, in particular to impose the sequential constraints on self assembly needed to produce the desired computation. In contrast to classical computing paradigms, the phenomena of self assembly create a randomness in the time required to perform a given computation. The research presented here lies in characterizing stochastic computing times within computational paradigms based on self assembly.

The notion of computing with tiling systems originated with Wang [14] over 55 years ago, but the modern theory applied to molecular computing emerged within the last decade in the work of Winfree [16], and Rothmund and Winfree [13]. The early theoretical work on general computation focused chiefly on various measures of complexity, in particular, program size and time complexity [1, 13, 15], but Adleman et al. [1, 2] investigated interesting combinatorial questions such as the minimum number of tile types needed for universality, and stochastic optimization questions such as the choice of concentrations that leads to minimum expected assembly times.

Our brief commentary has focused on mathematical modeling and anal-



Figure 1: The set of four tiles on the left implements the operation \oplus . A tile glues to other tiles only if their corner labels match. On the right operation $0 \oplus 1 = 1$ is performed. The input tiles are preassembled and the correct output tile simply attaches itself to the pattern, effectively obtaining the value of the output bit.

ysis of general DNA computation, as opposed to constructs solving specific combinatorial problems. We refer the reader to [2] for an extensive bibliography.

2.2 Growth models

In a reference theory for self assembly, it is natural to take the times between successive tile placements as independent, exponential random variables; for convenience, we take the mean as the unit of time. As described, the tiling assemblies of the last section are growth processes, although the computations themselves are processes of coalescence which terminate in a single value at the root of the template. As we shall see, it is more useful to focus on the equivalent growth times. We introduce next an abstraction that relates the times to grow (self assemble) constructs or patterns to classical theories of particle processes; growth is again subject to rules analogous to those governing the self assembly process of the previous section. The rectangle in the integer lattice defined by positions $(1,1)$ and (M, N) will be the model of self assembly studied here; it is a standard construct for this purpose, although it often further simplified to an $N \times N$ square.

An initial set of tiles (input) is placed along both coordinate axes, and growth proceeds outward in the positive quadrant: the placement of a new tile is allowed only if there are already tiles to the left and below the new tile's position which match labels as before. The left-and-below constraint is equivalent to requiring a newly added tile to bond at both of its input labels.

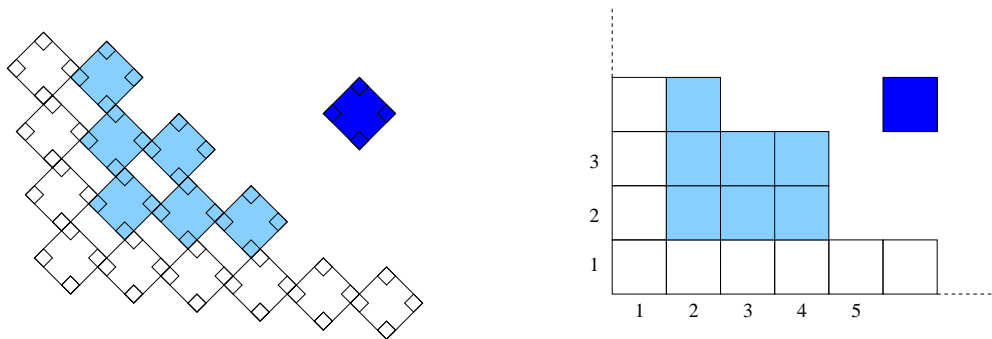


Figure 2: Growth models of DNA-based computation. The actual process of self assembly (left) can be equivalently represented as a growth process (right). There is a one-to-one correspondence between tiles on the left and squares on the right. White tiles (squares) represent the input.

(See Figure 2.) The eventual output of the computation can be thought of as the upper right corner tile at position (M, N) which can be attached only after all other tiles are in place.

The fundamental quantity of interest is the computation time, or equivalently, the time until this final, corner position becomes occupied by a tile. We let $C_{i,j}$ denote the time until the square (i, j) becomes occupied, so the random completion time is given by $C_{M,N}$. Let $T_{i,j}$ be the time it takes for a tile to land at position (i, j) once the conditions are favorable; that is, once both positions $(i, j - 1)$ and $(i - 1, j)$ are tiled. Recall that our basic assumption is that the $T_{i,j}$'s are independent exponential random variables with unit means. Then the completion times can be written in terms of attachment times $T_{i,j}$ by means of the recursion

$$C_{i,j} = \max(C_{i-1,j}, C_{i,j-1}) + T_{i,j},$$

with initial conditions $C_{i,1} = 0$ and $C_{1,j} = 0$, $1 \leq i \leq N$, $1 \leq j \leq M$. The recursion simply reflects the fact that a tile can be attached only if tiles to its left and below are already in place. By unwinding the recursion one can write the completion time as

$$C_{M,N} = \max_{\pi: (1,1) \rightarrow (M,N)} \sum_{(i,j) \in \pi} T_{i,j},$$

where the maximum is taken over all paths π from $(1, 1)$ to (M, N) consisting of segments going north or east only. We remark that the basic recursion

formula above can be thought of as an instance of an iterative multiplication of matrices in the so-called *max-plus* algebra, also known as *idempotent algebra* (see [11, 9]). In this formulation, the random values of interest, i.e., the completion times of a computation, are analogous to the matrix elements of the product of a large number of large random matrices. Alternatively, $C_{M,N}$ is the time when the M -th particle and N -th hole exchange positions in the totally asymmetric simple exclusion process (TASEP) on the integers starting from the megajam configuration: the positions (integers) to the left of the origin are all occupied by particles, and all positions to the right of the origin are empty. At independent, unit-mean exponentially distributed times, particles attempt to move to the right. A move actually occurs only if the adjacent position is a hole, i.e., is unoccupied. We further exploit the correspondence between the TASEP process and the assembly process on the plane in the next subsection.

Remarkably, the exact hydrodynamic-scale behavior of the TASEP process is known. As the rectangular DNA computer becomes large, or equivalently, as N, M grow to infinity such that M/N tends to a positive constant, one has [10, p. 412]

$$\lim_{N \rightarrow \infty} \frac{C_{M,N}}{(\sqrt{M} + \sqrt{N})^2} = 1. \quad (1)$$

The preceding formula quantifies the degree of parallelism in the computation. For example, let the required computation time be 1 second and let the template consist of $103 \times 103 = 10609$ tiles. Then, given that the attachment times are i.i.d. exponential random variables their common mean needs to be $1/(4 \cdot 103) = 0.25$ milliseconds.

We conclude this section with two observations. First, we have assumed that, prior to the computation, the set of input tiles needs to be prefabricated. Potentially, the input can be linear in N and, therefore, the prefabrication of the input might be the actual bottleneck of the computation. Second, our analysis assumes that no errors occur in the process of computation, i.e., a tile never bonds to an incorrect place on the template. While the main trade-off in microelectronic circuits is between the speed of computation and power consumption, it appears that in DNA-based computational devices it is between the the speed of computation and the correctness of the output. In particular, higher speed requires smaller attachment times, i.e., higher molecular mobility, which can result in higher error rates. In the next subsection we deal with the constraint of prefabricated borders. In the final

section, we consider a technique for recovering from errors.

2.3 Slowly growing borders.

It is now convenient to introduce continuous position variables x and y and argue in the hydrodynamic limit corresponding to (1)

$$C_{x,y} := \lim_{n \rightarrow \infty} \frac{1}{n} C(\lfloor nx \rfloor, \lfloor ny \rfloor)$$

for which [12]

$$C_{x,y} = (\sqrt{x} + \sqrt{y})^2 \tag{2}$$

and the path corresponding to this maximum duration is a ray from the origin to the point (x, y) . To be true for all (x, y) , this result requires the assumption that the ratio of border-tile to rule-tile concentrations is at least 1. However, this is not the case when an actual experiment is performed. These experiments suggest that the concentration of border tiles should be adjusted to be about half that of rule tiles. Experiments and simulation show that the profile of the self-assembly structure is triangular. We can modify (2) in order to predict the profile of the self-assembly structure under this new stoichiometry condition. Normalize the rule-tile growth rate to 1 and let α denote the border growth rate.

On calculating paths to a target (x, y) along which total tile attachment time in the fluid limit is maximum, one finds, with little effort, that to reach (x, y) :

- If α is sufficiently large ($\alpha \geq 1$ is always sufficient), then the path will be a straight line as before (see the path to (x_2, y_2) in Figure 3).
- If α is sufficiently small and $x < y$ then a path like ba in Figure 3 must be traversed. Specifically, near the beginning of self assembly, when the process is still relatively far from the target, the times taken to grow along border segments will dominate the maximum of times to the target along other initial path segments. With z as defined in the figure, the path of maximum duration will start from the origin and go to position $(0, y_1 - z)$ then from there go directly to (x_1, y_1) , so, speaking in terms of the discrete process, the time until a tile attaches

at position (x_1, y_1) is the sum of the times for tile attachments along the border path segment b and then along the direct path a .

- If α is sufficiently small and $x > y$ then we have the complementary path dc illustrated in Figure 3; the discussion above applies mutatis mutandis.

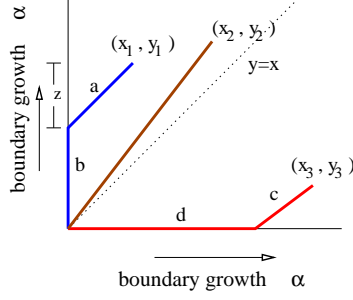


Figure 3: Illustrative paths to points (x_i, y_i) with maximum expected duration.

The maximum sum of the times required along paths like ba is

$$C_{x,y} = \sup_z \left((\sqrt{x} + \sqrt{z})^2 + \frac{y-z}{\alpha} \right) \quad (3)$$

A calculation then shows that

$$\begin{aligned} C_{x,y} &= x \left(1 + \frac{\alpha}{1-\alpha} \right)^2 + \frac{y}{\alpha} - \frac{x\alpha}{(1-\alpha)^2} \\ &= \frac{y}{\alpha} + \frac{x}{1-\alpha} \end{aligned} \quad (4)$$

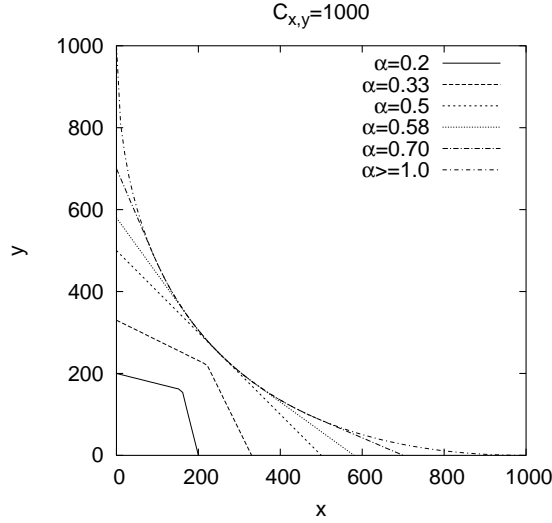
where

$$z = x \left(\frac{\alpha}{\alpha-1} \right)^2 \text{ and } z < y$$

An analysis of the paths like dc , when $x > y$, follows the same arguments and yields the same result with x and y interchanged.

Now fix the expected computing time (ECT), and plot iso-ECT curves, i.e., with α a parameter, plot coordinates (x, y) for which the asymptotic

expected self assembly time is the same. These curves are shown in Figure 4; under the curves is the structure that self assembles during the given expected time to reach any point on the curve.



(a)

Figure 4: Iso-ECT curves for various α at time $t = 1,000$

For $0 \leq \alpha \leq 1/2$, the curves are piece-wise linear (with a single articulation point on the diagonal) and lie below the straight line which corresponds to $\alpha = 1/2$. When $1/2 < \alpha < 1$, the curves are those above the straight line; the curves are linear near the border of the growth, then have the shape of $(\sqrt{x} + \sqrt{y})^2 = const$ sufficiently far from the borders. The latter shape applies to the entire curve when $\alpha \geq 1$.

3 The Pulsing Technique for Error Tolerance

A natural approach to error tolerance consists of *checkpoint restorative procedures*, a concept originating in classical computing. The idea is simple: to

return the computation periodically, when needed, to the latest stage (self-assembled structure) that was demonstrably error free.

In the nomenclature of the methods to reduce or eliminate errors in self-assembly, it is important to distinguish between *error detection* and *error correction*. It would be safe to say that virtually all methods of error detection envisioned now are centered on redundant computing. Indeed, finer tuned detection methods or an overall drastic decrease of error rates would require a dramatic change of the underlying mechanisms of self-assembly. Hence not much, within the existing paradigm, can be changed in the detection process.

Error correction can be done in many different ways. Redundant computation allows one to reduce error rates exponentially at a linear cost, but increases the size of the assembly by necessity, and thus might be of limited use. Further, the more complicated tiling systems are more error prone, with increasing ligation probabilities.

Checkpointing adds a crucial new component to all prior methods employed for error correction: the actual purging of the part of the assembly which is in error is performed in an active, controllable way. This allows one to optimize the level of accuracy of the error checking, at the same time maintaining the right balance between the speed and reliability of the process. We investigate below the trade-offs of speed versus reliability for the checkpointing process.

This process fundamentally depends on the parameter τ giving the time between successive tests for validity of the current self assembly. Other parameters (which may or may not be of significance in particular schemes) include the times taken by tests and by error-tile removal, whether this purging happens synchronously or not, etc.

The process of removing error tiles can be based on either or both of two general approaches. First, it can be *thermodynamically* based, exploiting the weaker, less stable bonds between error tiles and the adjacent structures. When heated, these bonds break down relatively easily, causing the error tiles to separate from the structure, along with the corrupt substructure seeded by the error tile, e.g., the quadrant up and to the right of the error tile in rectangle self assembly. Alternatively, checkpointing can be based on identifying the structural anomalies that accompany error tiles; accessing and removing the undesired structures can then be done by various means, e.g., enzymatically or using some kind of DNA-powered constructs, see e.g.[17].

The process of partial disassembly resulting from error tiles is illustrated in Figure 5. At the times of removal, the error tiles at the lower left corners

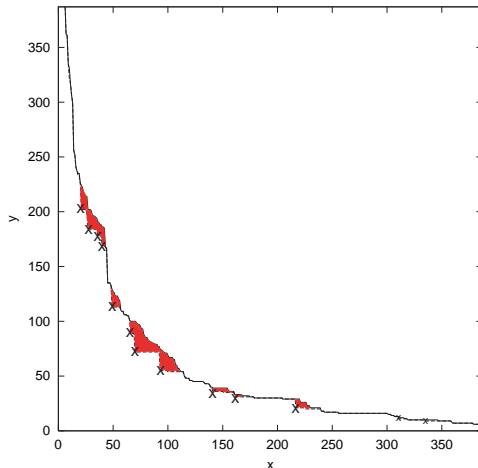


Figure 5: The state at a checkpoint. The shaded areas are corrupt regions seeded by error tiles marked by crosses

of corrupt regions are called *seed* error tiles. The seed error tiles appearing in one checkpointing stage define the shape of the structure beginning the next stage. There is a remarkable connection between the limiting number of checkpointing steps necessary for an error-free assembly (in the limit of large τ) and the problem of estimating the longest increasing subsequence in a random permutation (the Hammersley process) [6].

Suppose the desired self-assembled structure is an $N \times N$ square, and let ρ be the rate of error tiles, as before. In an appropriate scaling, where ρN^2 and τ are large and ρ is small, the pattern of seed error tiles is approximately Poisson with intensity ρ (the Principle of Deferred Decisions is at work here, as tiles are successively generated and corrupt ones removed in the checkpointing process). The expected total computation (self-assembly) time C_N is then easily seen as being proportional to the expected length of a longest increasing subsequence beginning at the seed tile in a Poisson pattern (‘increasing’ here means simultaneously in both dimensions). Thus, applying the classical result for the Hammersley process with the expected number ρN^2 of points (seed error tiles), we have the asymptotic result

$$C_N \sim 2(\rho N^2)^{1/2} = 2\rho^{1/2}N$$

as $N \rightarrow \infty$. Simulations have shown remarkably good agreement with this estimate in the appropriate parameter space.

References

- [1] L. Adleman, Q. Cheng, A. Goel, and M.-D. Huang. Running time and program size for self-assembled squares. In *Proc. ACM Symp. Th. Comput.*, pages 740–748, 2001.
- [2] L. Adleman, Q. Cheng, A. Goel, M.-D. Huang, D. Kempe, P. Moisset de Espanés, and P. Rothmund. Combinatorial optimization problems in self-assembly. In *Proc. ACM Symp. Th. Comput.*, pages 23–32, Montreal, Canada, 2002.
- [3] L. Adleman, Q. Cheng, A. Goel, M.-D. Huang, and H. Wasserman. Linear self-assemblies: Equilibria, entropy, and convergence rates. In Elaydi, Ladas, and Aulbach, editors, *New progress in difference equations*. Taylor and Francis, London, 2004.
- [4] Y. Baryshnikov, E. Coffman, and P. Momčilović. Incremental self-assembly in the fluid limit. In *Proc. 38th Ann. Conf. Inf. Sys. Sci.*, Princeton, NJ, 2004.
- [5] Y. Baryshnikov, E. Coffman, and P. Momčilović. Phase transitions and control in self assembly. In *Proc. Foundations of Nanoscience: Self-Assembled Architectures and Devices*, Snowbird, UT, April 2004.
- [6] Y. Baryshnikov, E. Coffman, N. Seeman, and B. Yimwadsana. Self correcting self-assembly: Growth models and the Hammersley process. In *Proc. 11th International Meeting on DNA Computing*, London, Canada, 2005.
- [7] Martin Feinberg. Lectures on chemical reaction networks. Autumn 1979.
- [8] D. Gillespie and L. Petzold. *System Modelling in Cellular Biology*. MIT Press, 2006. Z. Szallasi and J. Stelling and V. Perival, editors, Chapter title: Numerical Simulation for Biochemical Kinetics.
- [9] J. Gunawardena. An introduction to idempotency. In *Idempotency (Bristol, 1994)*, volume 11 of *Publ. Newton Inst.*, pages 1–49. Cambridge Univ. Press, Cambridge, 1998.
- [10] T. M. Liggett. *Interacting Particle Systems*. Springer-Verlag, New York, 1985.

- [11] G.L. Litvinov and V.P. Maslov. The correspondence principle for idempotent calculus and some computer applications. In *Idempotency (Bristol, 1994)*, volume 11 of *Publ. Newton Inst.*, pages 420–443. Cambridge Univ. Press, Cambridge, 1998.
- [12] H. Rost. Nonequilibrium behavior of a many particle process: Density profile and local equilibrium. *Z. Wahrsch. Verw. Gebiete*, 58:41–53, 1981.
- [13] P. Rothmund and E. Winfree. The program-size complexity of self-assembled squares. In *Proc. ACM Symp. Th. Comput.*, pages 459–468, 2001.
- [14] H. Wang. Proving theorems by pattern recognition, II. *Bell System Technical Journal*, 40:1–42, 1961.
- [15] E. Winfree. Complexity of restricted and unrestricted models of molecular computation. In R. Lipton and E.B. Baum, editors, *DNA Based Computing*, pages 187–198. Am. Math. Soc., Providence, RI, 1996.
- [16] E. Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, Pasadena, CA, 1998.
- [17] B. Yurke, A.J. Turberfield, Jr. A.P. Mills, F.C. Simmel, and J.L. Neumann. A DNA-fueled molecular machine made of DNA. *Nature, (London)*, 406(6796):605–608, 2000.