Running Time and Program Size for Self-assembled Squares*

Leonard Adleman† Qi Cheng‡ Ashish Goel§ Ming-Deh Huang¶

University of Southern California

Abstract

Recently Rothemund and Winfree [6] have considered the program size complexity of constructing squares by self-assembly. Here, we consider the time complexity of such constructions using a natural generalization of the Tile Assembly Model defined in [6]. In the generalized model, the Rothemund-Winfree construction of $n \times n$ squares requires time $\Theta(n \log n)$ and program size $\Theta(\log n)$. We present a new construction for assembling $n \times n$ squares which uses optimal time $\Theta(n)$ and program size $\Theta(\frac{\log n}{\log \log n})$. This program size is also optimal since it matches the bound dictated by Kolmogorov complexity. Our improved time is achieved by demonstrating a set of tiles for parallel self-assembly of binary counters. Our improved program size is achieved by demonstrating that self-assembling systems can compute changes in the base representation of numbers. Self-assembly is emerging as a useful paradigm for computation. In addition the development of a computational theory of self-assembly promises to provide a new conduit by which results and methods of theoretical computer science might be applied to problems of interest in biology and the physical sciences.

1 Introduction

Self-assembly is the ubiquitous process by which objects autonomously assemble into complexes. Nature provides many examples: Atoms react to form molecules. Molecules react to form crystals and supramolecules. Cells sometimes coalesce to form organisms. It has been suggested that self-assembly will ultimately become an important technology, enabling the fabrication of great quantities of intricate objects such as computer circuits from inexpensive components such as DNA and inorganic nanocrystals. Despite its importance, self-assembly is poorly understood. Recently, work related to DNA computation has led to experimental systems for the investigation of self-assembly and its relation to computation [8, 5, 3, 2]. In addition certain theoretical aspects of self-assembly have been considered. Winfree [8, 9] proved that self-assembling tile systems in a plane are capable of doing universal computation, and when restricted to a line are exactly as powerful as discrete finite automata. Adleman [1] proposed a mathematical model of self-assembly and analyzed the time complexity of linear polymerization. Rothemund and Winfree [6] proposed the Tile Assembly Model of self-assembly and studied the program size complexity (the number of different tile-types used) of constructing $n \times n$ squares. In this paper we extend the Tile Assembly Model to include the time complexity of the assembly process. We then demonstrate a system of tiles which assembles into an $n \times n$ square while simultaneously achieving optimal $(\Theta(n))$ time and optimal program size $(\Theta(\log n/\log\log n))$. In contrast, the system proposed by Rothemund and Winfree takes time $\Theta(n \log n)$ and uses $\Theta(\log n)$ program-size in

*A preliminary version of this paper will appear in the 33rd ACM Symposium on Theory of Computing, 2001.

†Professor of Computer Science and Molecular Biology, University of Southern California. Research supported by grants from NASA/JPL, NSF, ONR, and DARPA. Email: adleman@usc.edu

‡Department of Computer Science, University of Southern California. Research supported by NSF Grant CCR-9820778. Email: qcheng@cs.usc.edu

§Assistant Professor of Computer Science, University of Southern California. Email: agoel@cs.usc.edu

¶Professor of Computer Science, University of Southern California. Research supported by NSF Grant CCR-9820778 Email: huang@usc.edu

the worst case. It is our hope that an understanding of simple self-assembling systems will pave the way for a general theory of self-assembly.

To achieve the improved time complexity, we show how a binary counter that counts from 0 to n can be assembled in expected time $\Theta(n)$, as opposed to the $\Theta(n \log n)$ time for the same assembly in the work of Rothemund and Winfree. Further, the assembly time has an exponentially decaying tail. The number of increment steps required to count from 0 to n is exactly n. Thus our construction takes an amortized time of $\Theta(1)$ for an increment step, even though each increment step requires an addition of $\Theta(\log n)$ new tiles. This is made possible by the parallelism inherent in our tile-assembly system. We also observe that no system can result in a square being assembled in expected time less than $\Omega(n)$ in the model suggested by Rothemund and Winfree and expanded by us in this paper.

In order to count to n, a $\log n$-bit counter is required. Most of the $\Omega \log n)$ tiles in the construction of Rothemund and Winfree were used to produce the first row (the seed row) in the counter. The increment steps in their counter assembly (and in ours) can be performed using a constant number of tiles, and the fully assembled counter, which is roughly a $\log n \times n$ rectangle can be completed into a square using a constant number of tiles. To eliminate the need to produce a seed-row that is $\log n$ bits long, we represent the number n in base $\Theta(\log n / \log\log n)$ using only $\Theta(\log n / \log\log n)$ digits. Thus if the counter were to use base $\Theta(\log n / \log\log n)$ rather than binary, only $\Theta(\log n / \log\log n)$ tiles would be required to assemble a counter. This observation immediately allows us to reduce the number of tiles (the program size) required to assemble a square to $\Theta(\log n / \log\log n)$, which matches the lower bound dictated by Kolmogorov complexity. Unfortunately, this program size reduction comes at the cost of increased assembly time. To remedy this, we introduce the notion of base conversion. We start out with a row of $\Theta(\log n / \log\log n)$ tiles representing a number between 0 and n in base $\Theta(\log n / \log\log n)$. We then convert this number into binary using a self-assembly process that simulates base conversion. Now the counting process can take place in binary, allowing us to achieve the optimal time complexity and optimal program-size complexity *simultaneously*.

Section 2 presents a natural extension of the tile-assembly model of Rothemund and Winfree to include the time-complexity of self-assembly. Section 3 shows how a $\log n \times n$ counter can be assembled in time $\Theta(n)$ and section 4 explains how base conversion can be simulated in the tile-assembly model. Section 5 sketches how an entire $n \times n$ square can be constructed using the tools outlined in sections 3 and 4. Finally, we present some open problems and conclusions in section 6.

2 Adding time complexity to the Tile Assembly Model

The Tile Assembly Model was originally proposed by Rothemund and Winfree [6]. It extends the theoretical model of tiling by Wang [7] to include a mechanism for growth based on the physics of molecular self-assembly. Informally each unit of an assembly is a square with glues of various types on each edge. The tile "floats" on a two dimensional plane and when two tiles collide they stick if their abutting sides have compatible glues.

Formally, a tile is an oriented unit square with the north, east, south and west edges labeled from some alphabet Σ of glues. We begin with a triple $< T, g, \tau >$ where T is a finite set of tiles, $\tau \in \mathbf{Z}_{>0}$ is the *temperature*, and g is the *glue strength* function from $\Sigma \times \Sigma$ to $\mathbf{N}$, where Σ is the set of edge labels and $\mathbf{N}$ is the set of natural numbers. It is assumed that $null \in \Sigma$, $g(x, y) = g(y, x)$ for $x, y \in \Sigma$, and $g(null, x) = 0$ for all $x \in \Sigma$. For each tile $i \in T$, the labels of its four edges are denoted $\sigma_N(i)$, $\sigma_E(i)$, $\sigma_S(i)$, and $\sigma_W(i)$.

A *configuration* is a map from $\mathbf{Z}^2$ to $T \cup \{\mathbf{empty}\}$. For $t \in T$, $\Gamma_t^{(x,y)}$ is the configuration such that $\Gamma_t^{(x,y)}(i, j) = t$ iff $(i, j) = (x, y)$ and **empty** otherwise. Let C and D be two configurations. Suppose there exist some $i \in T$ and $(x, y) \in \mathbf{Z}^2$ such that $C(x, y) = \mathbf{empty}$, $D = C$ except at (x, y), $D(x, y) = i$, and

$$g(\sigma_E(i), \sigma_W(D(x+1, y)) + g(\sigma_W(i), \sigma_W(D(x-1, y)) +$$
$$g(\sigma_N(i), \sigma_W(D(x, y+1)) + g(\sigma_S(i), \sigma_W(D(x, y-1)) \geq \tau.$$

Then we say that the position (x, y) in C is *attachable*, and we write $C \rightarrow_{\mathbf{T}} D$ to denote the transition from

C to D in attaching tile i to C at position (x, y). Informally, $C \rightarrow_{\mathbf{T}} D$ iff D can be obtained from C by adding a tile to it such that the total strength of interaction in adding the tile to C is at least τ.

We define the notion of a *coordinated supertile* recursively as follows:

1. For $t \in T$, $\Gamma_t^{(x,y)}$ is a coordinated supertile.
2. if $C \rightarrow_{\mathbf{T}} D$ and C is a coordinated supertile, then D is also a coordinated supertile.

Two coordinated supertiles are equivalent iff they differ only by translation. An equivalence class of coordinated supertiles is an *supertile* (s-tile in brief). Write $A \rightarrow_{\mathbf{T}} B$ for s-tiles A and B iff there exist $a \in A$ and $b \in B$ such that $a \rightarrow_{\mathbf{T}} b$.

A *tile system* is a quadruple $\mathbf{T} =< T, S, g, \tau >$, where T, g, τ are as above and S is a set of supertiles called *seed supertiles.*

Let $\rightarrow_{\mathbf{T}}^*$ denote the reflexive transitive closure of $\rightarrow_{\mathbf{T}}$. A *derived supertile* of the tile system $\mathbf{T}$ is a supertile such that $s \rightarrow_{\mathbf{T}}^* A$ for some $s \in S$. A *terminal supertile* of the tile system $\mathbf{T}$ is a derived supertile A such that there is no supertile B, different from A, such that $A \rightarrow_{\mathbf{T}}^* B$. If there is a terminal supertile A such that for any derived supertile B, $B \rightarrow_{\mathbf{T}}^* A$, we say that the tile system *uniquely produces* A.

Given a tile system $\mathbf{T}$ which uniquely produces A, we say that the program size complexity of the system is $|T|$ i.e. the number of tile types.

In this paper, we adopt the restriction, suggested by Rothemund and Winfree [6], that S contains a single seed s consisting of a single tile, and that $g(\alpha, \beta) = 0$ for $\alpha, \beta \in \Sigma$ with $\alpha \neq \beta$. For a discussion of the lower bound on program-size in the absence of the latter restriction, see open problem #2 in section 6.

We now introduce the definition of the time complexity of self-assembly. A similar definition has also been suggested by Winfree [10]. We associate with each tile $i \in T$ a nonnegative probability p_i, such that $\sum_{i \in T} p_i = 1$. We assume that the tile system has an infinite supply of each tile, and p_i models the concentration of tile i in the system – the probability that tile i is chosen when a tile is drawn at random. Now self-assembly of the tile system $\mathbf{T}$ corresponds to a continuous time Markov process where the states are in a one-one correspondence with derived s-tiles, and the initial state corresponds to the seed s. There is a transition of state B to C iff $B \rightarrow_{\mathbf{T}} C$, and the rate of the transition is p_i if C is obtained from B by adding a tile i. Suppose the tile system uniquely produces an s-tile A_T. It would follow that A_T is the unique sink state. Given the Markov process, the time for reaching A_T from s is a random variable. The time complexity for producing A_T from s is defined as the expected value of this random variable.

Informally our definition of time models a system wherein a seed "floats" in solution encountering tiles at random. The higher the concentration of a particular tile the higher the rate at which it is encountered. When a tile is encountered which has sufficiently strong interaction with the seed, the tile is incorporated. By this process of accretion the seed grows larger and larger.

We say a tile system produces an $N \times N$ square iff it uniquely produces a terminal s-tile which is an $N \times N$ square of tiles. Then the time complexity of producing an $N \times N$ square is the minimum of the time complexity of all the tile systems which produce $N \times N$ squares. The following theorem is immediate from our formal model.

Theorem 2.1 *The time complexity of producing an $N \times N$ square is $\Omega(n)$.*

3 Counting up to n in time $\Theta(n)$

The square construction of Rothemund and Winfree [6] occurs in two stages. They first show how to assemble a $\log n \times n$ rectangle, and then extend it into an $n \times n$ square. To assemble the $\log n \times n$ rectangle, they simulate a counter that counts from 1 to n in binary. We are going to use the same general framework, but will replace their counter assembly by a more efficient (and more involved) process. In their counter assembly, only one tile is attachable to the assembly at any given time. In our assembly process, several tiles may be attachable at the same time. This "parallelism" allows us to assemble a counter in time $\Theta(n)$ as opposed to the $\Theta(n \log n)$ assembly time for the counter described by Rothemund and Winfree. We call the process described below the SA-counter.

3.1 The Tile System

We initially assume that $n = 2^K - 1$ for some positive integer K – we will later show how this assumption can be removed. While each tile is completely specified by its four glues, it is convenient for the purpose of exposition to allow tiles to have labels. Each tile has one main label (either 0 or 1) which corresponds to the bit represented by the tile. Further, each tile carries one or more auxiliary labels. The fully assembled counter is going to be a rectangle, with K tiles in each row. Each row represents a number between 0 and n – this number can be obtained by reading the main labels on the tiles in the row, with the most significant bit being the leftmost in the row. We describe the self-assembly process which results in the counter being incremented.

The increment operation happens in three stages and uses 15 different tiles (see figure 1). We start with an "INERT" row (each tile carries the auxiliary label "I"). This row gets replicated into an "ACTIVE" row (auxiliary label "A") if and only if there is at least one 0-tile in the original INERT row. If all the tiles in the INERT row are 1-tiles, the counter construction can not proceed any further and the self-assembly process terminates. A "CARRY" row (auxiliary label "C") then assembles on top of the ACTIVE row. The bulk of the increment operation happens during this step. The CARRY row will represent the value of the ACTIVE row incremented by one, except that the 0-tile (if any) which needs to change to a 1-tile due to a carry propagation is still unchanged. Then an INERT row assembles on top of the CARRY row which will convert such a tile (if any) to a 1-tile. The rightmost tile in any row always carries the auxiliary label "R" along with the auxiliary label corresponding to the row. Some tiles may also be marked special (auxiliary label "S") to aid in the carry propagation. Even though the above description seems serial, the SA-counter need not assemble row-by-row; several different tiles (possibly belonging to different rows) may be attachable at the same time.

We will assume that the temperature for this counter construction is 3. We now describe the tiles in each of the rows and their glues. All tiles occur with the same probability, p_{SA} which is a constant independent of n.

INERT tiles: There are the following INERT tiles: $0_{\mathrm{I}}, 1_{\mathrm{I}}, 0_{\mathrm{IR}}, 1_{\mathrm{IR}}, 0_{\mathrm{IS}}, 1_{\mathrm{IS}}$.

ACTIVE tiles: $0_{\mathrm{A}}, 1_{\mathrm{A}}, 0_{\mathrm{AR}}, 1_{\mathrm{AR}}$.

CARRY tiles: $0_{\mathrm{C}}, 1_{\mathrm{C}}, 0_{\mathrm{CS}}, 0_{\mathrm{CR}}, 1_{\mathrm{CR}}$. The glues and their strengths are depicted pictorially in figure 1.

Assembling ACTIVE rows: Notice that starting with an INERT row that consists of all 1-tiles, no ACTIVE tiles can attach to the top of the row since all possible bonds are of strength 2 whereas the temperature is 3. However, all 0-tiles in the INERT row can allow ACTIVE 0-tiles to attach on top through bonds of strength 3. These newly attached ACTIVE 0-tiles provide strength 1 bonds that supplement the strength 2 bonds between ACTIVE 1-tiles and INERT 1-tiles, allowing ACTIVE 1-tiles to attach in positions adjacent to the already attached ACTIVE 0-tiles. The newly attached ACTIVE 1-tiles in turn provide strength 1 bonds that allow adjacent ACTIVE 1-tiles to attach on top of INERT 1-tiles, and so on. This allows the entire INERT row to be replicated into an ACTIVE row.

Assembling CARRY rows: Notice that any 0-tiles (except the rightmost) in the ACTIVE row allow a CARRY 0-tile to attach on top by means of bonds of strength 3. These tiles then allow CARRY 1-tiles to attach on the left through bonds of strength 1, which in turn allow more CARRY 1-tiles to attach on the left. Notice that CARRY 0-tiles do not provide any glues on the right and therefore can not facilitate attachment of CARRY 1-tiles to their right. This allows most of the ACTIVE row to replicate into the CARRY row – we just need to bother about the rightmost tile and those ACTIVE 1-tiles that do not have an ordinary ACTIVE 0-tile to their right. We will now consider two cases depending on

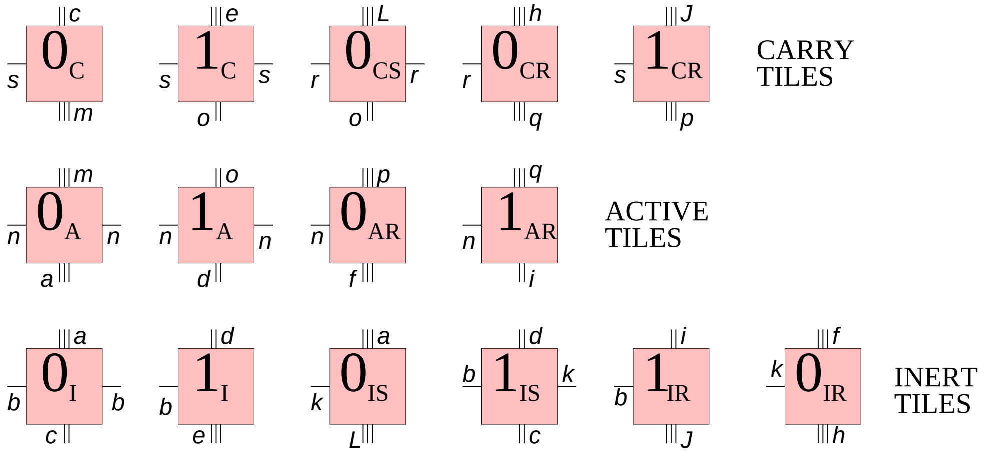


Figure 1: The tiles for the SA-counter. The number of lines jutting from each edge of the tile represent the strength of the bond (1,2, or 3) and the label on the edges represents the glue. Some edges do not have any glues.

the rightmost tile in the ACTIVE row. If the rightmost tile in the ACTIVE row is a 0-tile then it will allow a rightmost CARRY 1-tile to attach on its top through a bond of strength 3. This then provides a bond of strength 1 to its left which will allow ACTIVE 1-tiles to assemble to its left, thus completing the CARRY row. Notice that this CARRY row already represents the correct result for the increment operation. If the rightmost tile is an ACTIVE 1-tile, then it allows a rightmost CARRY 0-tile to attach on top. This tile then allows special CARRY 0-tiles (and not CARRY 1-tiles) to bond to its left on top of the ACTIVE 1-tiles. Thus the entire sequence of contiguous ACTIVE 1-tiles on the right will have special CARRY 0-tiles attached on top. The CARRY row now represents the correct result for the CARRY operation with one exception: the ACTIVE 0-tile which was immediately to the left of the rightmost sequence of ACTIVE 1-tiles has a CARRY 0-tile attached on top, whereas the increment operation should convert it into a 1-tile due to carry propagation. This is remedied in the next step.

Assembling INERT rows: The 0-tile which needs to be changed into a 1-tile can be "detected" as a CARRY 0-tile which has a special CARRY 0-tile or a rightmost CARRY 0-tile to its immediate right. The description for the assembly of the INERT row is simple. All the tiles except ordinary CARRY 0-tiles can attach an INERT tile (with the other labels such as the bit-label, the rightmost label, and the special label remaining the same) on top. Now special INERT 0-tiles and rightmost INERT 0-tiles allow a special INERT 1-tile to attach (on top of a CARRY 0-tile) to their left, while all other INERT tiles allow an INERT 0-tile to attach. This accomplishes the desired carry propagation and completes the increment process.

The Seed Row: The seed supertile S_K is a row of K special tiles. The rightmost tile in S_K is identical to the tile 0_{IR} except that the bottom surface has no glue. The other $K-1$ tiles are identical to the tile 0_{I} except that there is no glue on the bottom surfaces[1].

The tile system $\mathbf{T}_{SA}(K)$ has a tile set consisting of all the tiles in figure 1 as well as the special tiles needed in the supertile S_K. The glue strength function is as indicated in figure 1. The temperature is 3, and there is a single seed supertile S_K. We will refer to the Markov chain defined by the tile system as the SA-counter. We will assume that all tiles which are not in S_K have the same constant probability.

The next theorem follows from our description of the tile system, and we omit the proof.

[1] In this section we are using supertiles consisting of more than one tile as the seed, whereas our restricted model allows us to use only supertiles with a single tile as seed. We rectify this in section 5.

Theorem 3.1 *The tile system $\mathbf{T}_{SA}(K)$ uniquely produces a supertile which is a $K \times (3 \cdot 2^K - 2)$ rectangle.*

Minor modification of the seed row results in a tile system that uniquely produces a supertile which is a $K \times N$ rectangle for any $N \leq 3 \cdot 2^K - 2$.

Our counter construction is more involved than that proposed by Rothemund and Winfree [6] but exploits "parallelism" to speed up the assembly process. The construction of Rothemund and Winfree takes time $O(n \log n)$ in the model of running time described in Section 2. Define the chain-length, C, of a row as the maximum number of contiguous 1-tiles or contiguous 0-tiles in the row. Once a row gets assembled in our construction, each tile in the next row depends on at most $O(C)$ more tiles attaching before it becomes attachable. In the construction of Rothemund and Winfree, a tile in the next row may have to wait for $\log n$ tiles to attach (in serial) before it becomes attachable, even if C is small. This distinction is the main source of parallelism in our construction. The average chain length is $O(\log \log n)$ as the counter construction proceeds from 1 to n. Combining this with Chernoff bounds, we can immediately conclude that the average expected time for an increment operation is $O(\log \log n)$ which yields an $O(n \log \log n)$ upper bound on the assembly time of our counter. This is not the best bound we can prove (we show that SA-counter gets assembled in linear time in section 3.2). Intuitively, the stronger $\Theta(n)$ bound on the assembly time is due to the fact that a row may start getting assembled even before the previous row finishes getting assembled.

3.2 Analysis

For the purpose of our analysis, we transform the SA-counter into another process which we call the "sentinel" process. The sentinel process does not adhere to the model described in Section 2; in fact there does not seem to be an easy implementation of the sentinel process. However, the sentinel process is more amenable to analysis, and the time for this process to complete is an upper bound (in the stochastic domination sense) on the completion time for the SA-counter.

The sentinel process: Look at a completed self assembly, and replace each bond by two directed bonds, one in each direction. Each directed bond has the same strength as the original bidirected bond. Then, remove all the directed bonds that satisfy any of the following criteria:

1. The bond goes from a higher to a lower row.
2. The bond goes from left to right in a non-ACTIVE row.
3. The bond goes from left to right in an ACTIVE row, and there is at least one zero-tile to the right of the origin of the bond.
4. The bond goes from right to left in an ACTIVE row, and the destination tile of the bond is a zero-tile.
5. The bond goes from right to left in an ACTIVE row, the source and destination tiles are both one-tiles, and there are no zero-tiles to the right of the source.

We can define a sentinel graph as the graph with vertices corresponding to the tiles, and directed arcs corresponding to the directed bonds and labeled by the strength. The following lemma is immediate from our construction.

Lemma 3.2 *The sentinel graph is acyclic.*

The sentinel process is a Markov chain obtained by modifying the Markov chain corresponding to the SA-counter as follows. We consider each transition in the Markov chain for the SA-counter. Let this transition correspond to adding a new tile X to a supertile A. A tile Y in A is said to be a *support tile* if it shares a side with X and there is an arc from Y to X in the sentinel graph; the strength of this arc is said to be the *support strength* from Y. We retain this transition if the sum of the support strengths from the support tiles is greater than the temperature and else we discard it. Any state in the Markov chain that is unreachable from the source state is discarded.

Intuitively, the sentinel process is formed by taking the SA-counter and introducing a "sentinel" who disallows transitions that *require* bond-formation from the left to the right, except when such transitions are necessary for replication. It is not difficult to see that the sentinel process produces exactly the same complete assembly as the SA-counter.

In the sentinel graph, there is exactly one bond that goes from the leftmost column to the right, and this happens just above the INERT row $011\ldots11$ where the leftmost 0 is the only tile that can "self-replicate" into the active row and must then induce the 1-tiles to attach to its right. Further, this property is recursively satisfied if we remove the leftmost column and look at the sentinel graphs for the two sub-rectangles below and above the $011\ldots11$ row. This recursive structure makes the sentinel process more amenable to analysis. The structure of the sentinel graph is illustrated in figure 2.

Stochastic dominance: Define t_{ij} to be the time at which the (i,j) position gets filled in the SA-counter and t'_{ij} to be the time at which it gets filled in the sentinel process. Note that $t(i,j)$ and $t'(i,j)$ are random variables. Let $t(n)$ and $t'(n)$ be the random variables denoting the times at which the SA-counter and the sentinel assembly complete.

A real valued random variable A is said to be stochastically dominated by another random variable B, denoted $A \leq_{sd} B$, if for all x, $\mathbf{Pr}[A > x] \leq \mathbf{Pr}[B > x]$.

Lemma 3.3 *For all position (i,j) in the constructed counter, $t_{ij} \leq_{sd} t'_{ij}$*

Proof: Let X_{ij} be an exponential random variable with mean $1/p_{ij}$, where p_{ij} refers to the probability associated with the tile at position i,j in the square. Recall that $p_{ij} = p_{SA}$ for all i,j, where p_{SA} is some constant independent of n. Let all the X_{ij} be independent. A tile attaches at position (i,j) in the self-assembly X_{ij} time after this position becomes attachable[2]. We couple the sentinel process and the SA-counter by setting the values of X_{ij} to be the same for both processes. Define a_{ij} and a'_{ij} to be the times at which tile position (i,j) is attachable in the SA-counter and the sentinel process, respectively. Let t be the earliest time when a tile gets attached in the sentinel process but is still unattached in the SA-counter. Let (i,j) be this tile position. Clearly, $a'_{ij} < t$. Therefore any tiles which had attached in the sentinel process by time a'_{ij} had also attached in the SA-counter. Since the sentinel process was formed by *disallowing* certain bonds in the SA-counter, tile position (i,j) is also attachable in the SA-counter at time a'_{ij}. Hence $a_{ij} \leq a'_{ij}$. But $t_{ij} = a_{ij} + X_{ij}$ and $t'_{ij} = a'_{ij} + X_{ij}$. This implies that $t_{ij} \leq t'_{ij}$, which is a contradiction. Since $t_{ij} \leq t'_{ij}$ for each coupled experiment, $t_{ij} \leq_{sd} t'_{ij}$. ■

Completion time for the sentinel process: Define $L(n)$ to be the longest length of any directed path in the sentinel graph for counting up to n.

Lemma 3.4 *$L(n) = \Theta(n)$.*

Proof: Let us break the sentinel graph into three parts to take advantage of the recursive structure of this graph. The first part corresponds to all the rows from the initial all-zeros INERT row to the INERT row that has a 0 in the most significant position and a 1 everywhere else. The second part corresponds to the increment operation on this INERT row. The third corresponds to all the remaining rows. Let $G_1(n), G_2(n), G_3(n)$ refer to these three parts of the sentinel graph and let $G(n)$ refer to the entire graph. The recursive structure of the graph is depicted in figure 2. Let $L_1(n)$, $L_2(n)$, and $L_3(n)$ represent the maximum length of any path in each of the three parts. Since no bonds go from higher to lower rows (by construction of the sentinel graph), we have $L(n) \leq 2 + L_1(n) + L_2(n) + L_3(n)$. Since the graph is acyclic and there are only $O(\log n)$ tiles in the second part, $L_2(n) = O(\log n)$. The first and third parts of the graph are symmetric so we will just concentrate on $L_1(n)$. The crucial observation is that the leftmost tile (a 0-tile) in the inert row with which

[2]This fact follows from the fact that once a tile position becomes attachable it remains attachable till it actually attaches.

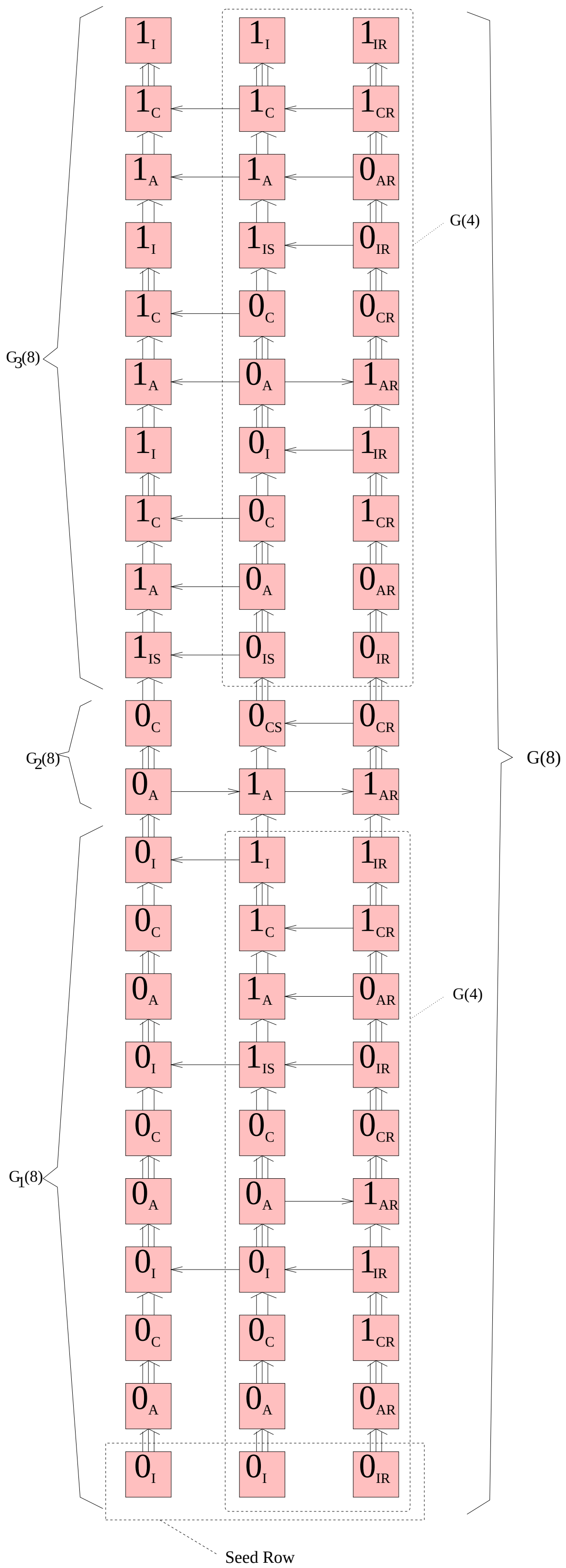


Figure 2: The recursive structure of the sentinel graph.

the first part terminates is the tile at which any longest path in the entire first part must end. Further, the structure of the sentinel graph dictates that the length of the longest path ending at this tile is $L(n/2)+1$. Therefore, $L(n) = 2L(n/2) + \Theta(\log n)$, and the solution to this recurrence is $L(n) = \Theta(n)$. ∎

Recall that $t(n), t'(n)$ are the completion times for the SA-counter and the sentinel process respectively.

Lemma 3.5 $\mathbf{E}[t'(n)] = \Theta(n)$. *Further, $t'(n)$ has an exponentially decaying tail.*

Proof: Let $P_1, P_2, \ldots P_N$ represent the N directed paths from the seed row to any tile in the final row in the sentinel graph. At each step in any path, the bond must go either up, left, or right. Therefore N is at most $3^{L(n)} \leq e^{cn}$ for some constant c. Let S_l denote the sum of all X_{ij} such that position ij lies on path P_l. Then the completion time $t'(n) = \max_{l=1}^{n} S_l$. S_l is the sum of at most $L(n)$ mutually independent exponential variables, each with mean $1/p_{SA}$. Hence $\mathbf{E}[S_l] \leq L(n)/p_{SA}$; let ϕ denote the value $L(n)/p_{SA}$. Clearly $\phi = O(n)$. Using Chernoff bounds for exponential variables [4], it follows that $\mathbf{Pr}[S_l > \phi \cdot (1+\delta)] \leq ((1+\delta)/e^{\delta})^{L(n)} \leq ((1+\delta)/e^{\delta})^n$. Hence $\mathbf{Pr}[t'(n) > \phi(1+\delta)] \leq N \cdot ((1+\delta)/e^{\delta})^n \leq e^{cn}((1+\delta)/e^{\delta})^n = ((1+\delta)/e^{\delta - c})^n$. Let us choose $\delta = \delta' + 2\max\{c, 2\}$, where $\delta' > 0$. Now

$$\begin{aligned}
&\mathbf{Pr}[t'(n) > \phi(1+2\max\{c,2\})(1+\delta')] \\
&\leq \mathbf{Pr}[t'(n) > \phi(1+2\max\{c,2\}+\delta')] \\
&\leq ((1+\delta')/e^{\delta'})^n.
\end{aligned}$$

This clearly gives an exponential tail bound. Now,

$$\mathbf{E}[t'(n)] \leq \phi(1+2\max\{c,2\})(1+\int_{\delta'=0}^{\infty} ((1+\delta')/e^{\delta'})^n d\delta'),$$

which is $O(n)$ as $\phi = O(n)$ and the integral is bounded by a constant for any value of $n \geq 1$. ∎

Lemma 3.3 in conjunction with the lemma 3.5 now allows us to conclude:

Theorem 3.6 $\mathbf{E}[t(n)] = \Theta(n)$. *Further, $t(n)$ has an exponentially decaying tail.*

4 Simulating base conversion by self-assembly.

In Rothemund and Winfree's construction (and in ours), the "seed row" represents a number written in binary and consists of $\log n$ tiles, each encoding 0 or 1. The number is used as the starting point for a counter. Since we are only allowed a seed supertile consisting of a single tile, this seed row itself needs to be assembled from a single tile. Each tile in the seed row needs to be distinct, and therefore the program size complexity for assembling the seed row itself is $\Omega(\log n)$. On the other hand, Kolmogorov complexity dictates the program size complexity to be $\Omega(\frac{\log n}{\log\log n})$. In order to achieve the Kolmogorov bound, we change the seed row to represent a number written base b, where b is power of 2 such that

$$\frac{\log n}{\log\log n} \leq b = 2^k < \frac{2\log n}{\log\log n},$$

for k a positive integer. Then the number of tiles to construct the seed row will be

$$h = \frac{\log n}{\log b} < \frac{\log n}{\log\log n - \log\log\log n} = O(\frac{\log n}{\log\log n}).$$

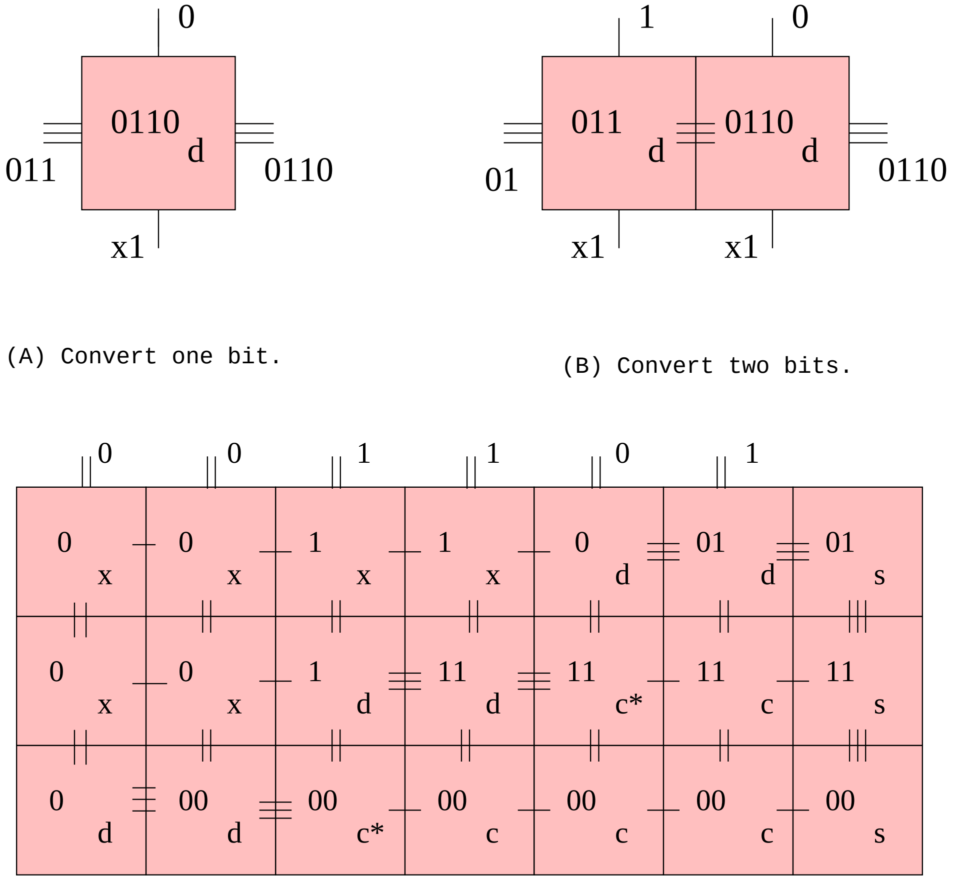
As in Rothemund and Winfree [6]'s counting scheme, we use the "seed row" as a starting point for a counter. If we use the counting strategy of Rothemund and Winfree, modified to count base b, we easily achieve the optimal program size complexity or in the language of Rothemund and Winfree:

Theorem 4.1 $\mathbf{K}^2_{SA}(N) = O(\frac{\log N}{\log\log N})$.

Unfortunately if we simply follow Rothemund and Winfree [6]'s method, we pay a price in time complexity. The Rothemund and Winfree construction begins with the seed row and grows through a succession of s-tiles to produce a final rectangle. But for each s-tile produced in the process, there is exactly one tile which can be incorporated. There are at least $b = \Theta(\frac{\log n}{\log\log n})$ distinct tiles which occur $\Omega(n)$ times in the final rectangle, and at least one of these tiles must have probability $O(\frac{\log\log n}{\log n})$. It follows that such tiles require $\Omega(\frac{n\log n}{\log\log n})$ time for assembling of the rectangle. In particular, the time is not linear. Even using the parallel construction described in section 3, the time is not improved.

To overcome this time problem while preserving the $O(\frac{\log n}{\log\log n})$ program size, we adopt the strategy of writing the seed row base b and then converting it to base 2. We then employ the fast parallel counter described in section 3.

In the conversion part, we will employ some tiles, which have the capability to perform division-by-2. These tiles have glues representing binary strings. The north glue represents last bit, the west glue represents the prefix substring and the east glue represents the string itself. See Figure 4 (A) for the tile associated with the string 0110, and Figure 4 (B) for cooperation of two tiles. We assume that the temperature is 3, but it is not hard to see that the same idea also works for temperature 2.



(C) Convert 031 in base 4 to 001101 in base 2.

Figure 3: Converting base by self-assembly.

In the following tile descriptions, we identify a tile by the labels on its four sides: North(N), West(W), East(E) and South(S), and a title symbol(TS). In our notations, a prefix "(2)" indicated a glue with strength 2, a prefix "(3)" indicated a glue with strength 3. The absence of prefix indicates glue with strength 1.

Suppose we want to convert $b_1 \cdots b_h$ in base b to binary base number, where b_i if written in binary would be $b_{i,1}b_{i,2}\cdots b_{i,k}$. The set of tiles consists of

1. (Seed.) These tiles will form a seed row. For each $b_{i,1}b_{i,2}\cdots b_{i,k}$, $1 \le i \le h$,

   TS: $(b_{i,1}b_{i,2}\cdots b_{i,k})_{\mathrm{s}}$
   N: if $i < h$, (3)S_i; S: If $i > 1$, (3)S_{i-1};
   W: if $i = h$, (3)$b_{h,1}b_{h,2}\cdots b_{h,k}$, else $b_{i,1}b_{i,2}\cdots b_{i,k}$.

2. (division by 2.) For each binary string $a_1a_2\cdots a_i$, $1 \le i \le k$,

   TS: $(a_1a_2\cdots a_i)_{\mathrm{d}}$,
   N: (2)a_i, E: (3)$a_1a_2\cdots a_i$,
   W: if $i \ge 2$, (3)$a_1a_2\cdots a_{i-1}$, otherwise x2,
   S: if $i \ge 2$, (2)x1, otherwise (2)x3.

3. (Base b copy.) For each binary string $a_1a_2\cdots a_k$,

   TS: $(a_1a_2\cdots a_k)_{\mathrm{c}}$,
   N: (2)x1, W: $a_1a_2\cdots a_i$, E: $a_1a_2\cdots a_i$, S: (2)x1.

4. (Last b copy tile.) For each binary string $a_1a_2\cdots a_k$,

   TS: $(a_1a_2\cdots a_k)_{\mathrm{C}*}$
   N: (2)x3, W: (3)$a_1a_2\cdots a_i$, E: $a_1a_2\cdots a_i$, S: (2)x1.

5. (Base 2 copy.) For $a = 0$ or 1,

   TS: a_{x}; N: (2)a; E: x2; S: (2)a; W: x2.

Note that the east side of the seed tiles, the south side of the first seed tile and the north side of the last seed tile are not assigned with any glues. They are open for later use. Even if those glues are all different, the number of distinct tiles is not increased.

Figure 4(C) is an example showing how to convert string 031 in base 4 to string 001101 in base 2.

In item 2, we use the most distinct tiles. which is

$$b + \frac{b}{2} + \frac{b}{4} + \cdots 1 < 2b = O(\frac{\log n}{\log\log n}).$$

Thus the number of distinct tiles in this conversion subroutine is $O(\frac{\log n}{\log\log n})$. Every tile appears at most $O(\log n)$ times. We can arrange that each tile has probability greater than $\Omega(\frac{\log^2 n}{n\log\log n})$, in which case, the time to do the conversion is at most $O(n)$.

5 Putting it all together

The self-assembly of an $n \times n$ square begins with a seed tile which grows into a seed row consisting of roughly $b = \Theta(\frac{\log n}{\log\log n})$ tiles representing a number m written base b. This seed row spawns a base conversion assembly as outlined in section 4 The result is a rectangle with top row representing m written base 2. This top row becomes the starting point for a binary counter assembly as outlined in section 3.

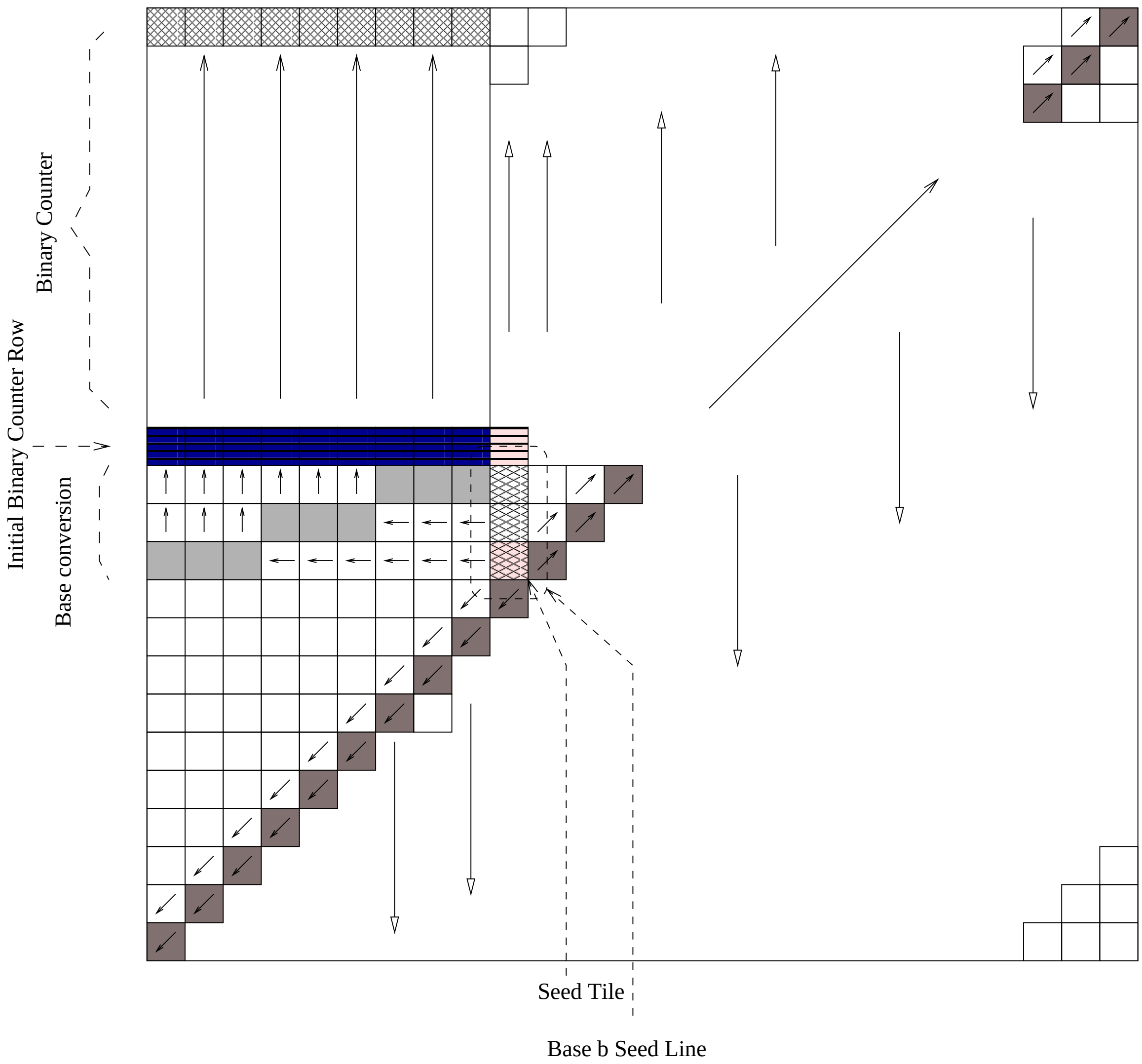


Figure 4: Constructing a square by self-assembly.

The height of the resulting counter rectangle is a function H of m. H satisfies $H(m+1) = H(m) + 3$. Hence by selecting an appropriate m, the base conversion assembly and the counter assembly have dimensions such that height plus width equals $n - e$, where $e = 0, 1$ or 2. We can grow e extra rows at the base of the combined rectangle to bring the sum of the two dimensions exactly up to n. The resulting rectangle acts as the basis for the completion of the $n \times n$ square as described in [6] using diagonal elements and filler tiles. Figure 4 describes this process pictorially.

By allocating a combined probability of 1/5 to the tiles constructing the seed row, 1/5 to the tiles which perform the base conversion, 1/5 to the tiles which perform the counter construction, 1/5 to the diagonal elements and 1/5 to the filler tiles, it follows that the time to construct the $n \times n$ square is $O(n)$. The fact that the $\Theta(n^2)$ filler tiles get assembled in time $O(n)$ can be proved in a manner similar to that used for Lemma 3.5.

6 Conclusions and open problems

We have generalized the Tile Assembly Model of Rothemund and Winfree [6] to include time complexity and have demonstrated a tile system for self-assembling $n \times n$ squares that is optimal in both time and program size. The long term goal of this research is the development of a mathematical-computational theory of self-assembly that will combine classical theories such as thermodynamics and statistical mechanics with modern theories such as combinatorics and computational complexity.

There are many open problems. For example:

1. Reducing the temperature τ of our construction to 2 rather than 3.

2. As in [6] we have assumed that $g(\alpha, \beta) = 0$ for $\alpha, \beta \in \Sigma$ with $\alpha \neq \beta$. Without this assumption, Kolmogorov complexity no longer dictates an $\Omega(\log n / \log \log n)$ lower bound on the program size (number of tiles). What is the smallest possible number of tiles needed in that case, and can this bound be achieved simultaneously with optimal program time? The Kolmogorov complexity based lower bound for this model is $\Omega(\sqrt{\log n})$ tiles.

3. In the model described here, s-tiles grow by accretion i.e. by the addition of single tiles to a growing crystal. Modify the model to allow any two s-tiles to merge into a larger s-tile. In such a system the lower bound of $\Omega(n)$ for the time to assemble $n \times n$ squares no longer appears necessary. In this case, what is the best possible time and can it be achieved simultaneously with optimal program size?

4. The Tile Assembly Model as presented here is "irreversible" - once a tile sticks, it never "unsticks". Generalize to a model that allows tiles to both attach to and detach from the assembly.

Acknowledgements

We would like to thank Paul Rothemund and Erik Winfree for several helpful discussions about their work [6].

References

[1] Leonard M. Adleman. Towards a mathematical theory of self-assembly. Technical Report 00-722, Department of Computer Science, University of Southern California, 2000.

[2] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman and R. Weiss. Amorphous computing. *AI Memo 1665*, August 1999.

[3] Chengde Mao, Thomas H. LaBean, John H. Reif and Nadrian C. Seeman. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature*, 407:493–496, 2000.

[4] R. Motwani and P. Raghavan. Randomized Algorithms. *Cambridge University Press, 1995.*

[5] Paul Rothemund. Using lateral capillary forces to compute by self-assembly. *Proceedings of the National Academy of Sciences*, 97:984–989, 2000.

[6] Paul Rothemund and Erik Winfree. The program-size complexity of self-assembled squares. *STOC 2000.*

[7] H. Wang. Proving theorems by pattern recognition. II. *Bell Systems Technical Journal*, 40:1-42, 1961.

[8] Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394:539-544, 1998.

[9] Erik Winfree. PhD thesis. Cal Tech, 1998.

[10] Erik Winfree. *Personal communication.*