Molecular Self-Assembly

Boonsit Yimwadsana

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2007
UMI Number: 3266706

Copyright 2007 by
Yimwadsana, Boonsit

All rights reserved.

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI Microform 3266706
Copyright 2007 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346
ABSTRACT

Molecular Self-Assembly

Boonsit Yimwadsana

The parallelism of molecular computation has made algorithmic self assembly one of the most exciting fields in the past decade. Chemical kinetics contributes to our understanding of self assembly through the models provided by systems of reaction rate equations in which the variables are time-dependent concentrations of the various self assembled structures. However, these systems of ordinary differential equations are usually non-linear and it is very difficult to find explicit solutions for them. In the case where there are a large number of distinct structures involved, a large number of equations are needed, and even numerical methods have limited usefulness. On the other hand, we have been able to solve for the final yield (the concentration of the desired self-assembled structures), once the system is in its absorbing state. This thesis provides novel methods for obtaining explicit formulas for the yield of simple systems [13, 11]. This analysis of yields led us to the discovery of previously unknown phase transition phenomena where 100% yields are obtained for certain subsets of the reaction rate parameters.

To acquire insights into more complex systems, we use numerical methods and simulations. In the latter case, we formulate discrete-event simulations from elementary collision theory which leads to a Markovian birth-and-death process that models self assembly events (see [17]). The novelty of our approach lies in its simplicity and in our proof that, in the hydrodynamic limit, the birth and death equations become reaction-rate equations of chemical kinetics.
These contributions form Part I of the thesis. Part II shifts to the algorithmic self assembly of specific molecules, viz., those constructed from DNA. Computation becomes a crystal-growth process in two dimensions; it is modeled formally by tiling systems whereby distinct tiles represent the various types of molecules; each tile has edge encodings that determine the types of molecules that can bond with the given tile during crystal growth, and which are determined by the computation desired of the tiling. Here, our interest is no longer in yields, but rather the time required to complete the computation implicit in crystal growth. Our key contribution is a mapping of this two-dimensional tiling process into a one dimensional particle process, and our discovery that it is precisely the well-known ‘totally asymmetric simple exclusion process’ (TASEP) widely studied by those in applied mathematics and the physical sciences. Once we discovered that event in the TASEP which was equivalent to the time to self assemble (i.e., tile) a rectangular structure, we could apply well known results from the TASEP literature. (See our work in [12, 11] which focuses on the asymptotics of hydrodynamic limits.)

These tiling results are based on idealized crystal-growth initial conditions. In the laboratory, these initial conditions are in fact part of the growth process, an observation that complicates the process substantially. Our entirely new analysis (see [15, 16]) of this generalized tiling process leads to explicit formulas for self-assembly times, once again in the hydrodynamic limit.

The low-energy, vast parallelism of molecular computation is perhaps its chief advantage over alternative paradigms. It is obvious interest therefore to quantify this parallelism by means of an analysis that yields explicit formulas. Just such formulas describing parallelism in the tiling model are among the original contributions of this thesis.
Errors in self assembly can and do occur in practice and to an extent that creates a serious design problem that must be taken into account. We propose and analyze a novel temperature pulsing method (see [14]). The method is based on a roll-back process analogous to the well-known checkpointing method of error tolerant computing systems. By periodic temperature pulses the weakly bonded erroneous regions created during crystal growth can be detached from properly self assembled regions of the crystal. Again, by a key observation related to the well-known Hammersley process, it is shown that the expected time to self assemble square tilings is proportional to the expected length of the longest increasing subsequences of random permutations of a given set of integers.
# Contents

## I Introduction

1 Overview  

2 Models of Self-Assembly Kinetics  

3 Tile Self-Assembly Models  

   3.1 Computation by Tile Self Assembly  

   3.2 Error-Correction in Tile Self-Assembly

## II Self Assembly Models

1 Introduction  

2 An Experimental Tool  

3 Analytical Insights  

   3.1 Self Assembly with $N = \infty$  

   3.2 Triangle Self Assembly  

4 Incremental Self Assembly  

   4.1 Assembly of Triangles  

   4.2 Assembly of Systems with Arbitrary $N$
IV  Appendix

Bibliography
Acknowledgement

This thesis would not have been possible without the support of many people. It is not possible to give sufficient thanks to my advisor, Edward G. Coffman, who read my numerous revisions and guided me through the process of doing research. I also would like to thank my mentor and collaborator, Yuliy Baryshnikov, who has been so wonderful in helping me through difficult problems both technical and non-technical. In many ways, my research can be seen as a direct extension of their work. Both Ed and Yuliy have provided encouragement, support and valuable feedback throughout this project. I would like to thank Columbia University for providing me excellent research facilities and colleagues. My life in school would not be fun and happy without the magnificent support from David, Symone, Cole and Jenna who help me with almost everything I would ever need. They are like a parent figure to me. Finally, I would like to thank my parents and my friends for every support and encouragement that motivate me through out my entire journey at Columbia.
Part I

Introduction
Chapter 1

Overview

Self assembly refers to the bottom-up paradigm of numerous constructive processes, both natural and artificial, whereby elementary particles (e.g., molecules) autonomously combine to create clusters of particles which in turn combine with other clusters to form progressively larger clusters. This thesis begins with studies of self assembly in arguably the simplest possible setting, one that falls within the classical theory of chemical kinetics. In particular, abstract, dimensionless models of self-assembly are the subject of Part II of this thesis, where the terms “particle” and “cluster” are replaced by “monomer” and “polymer” and thus made consistent with the early work of Adleman et al [1] on elementary polymerization models. Nothing about the theory itself relies on molecular scaling, e.g., it could apply to the self assembly of objects on the Internet [24]. However, the accuracy of our asymptotic estimates is established here only at relatively small scales. Notions of yields, self-assembly times, and phase transitions of self-assembly processes are briefly introduced in Chapter 2 below, which reviews the research described in Part II.

Historically, intense interest in self-assembly as a computational or algorithmic paradigm originated in Adleman’s famous experiment [4]. Adleman showed that a
CHAPTER 1. OVERVIEW

The parallelism of linear self-assembly using Adleman’s method can be used efficiently to evaluate Boolean formulas. For example, Adleman’s concept was used by Braich et al. [20] on the satisifiability problem. Adleman’s combined method of self-assembly and filtering was very specific to the Hamiltonian Path Problem. The filtering process is more difficult than the self-assembly process, so a lot of attention shifted towards the former process. Adleman’s work was generalized by Lipton [40, 41], who focused on formalizing the filtering method.

In 1998, Winfree [56] fit linear self-assembly into the conventional models of com-

---

1 For a given graph $G$, the Hamiltonian Path problem asks whether there exists a path in $G$ that visits each node exactly once.

2 bonding via the hydrogen bonds between two DNA strands.

3 covalent bonding which joins two DNA strands in the presence of the ligase enzyme.
CHAPTER 1. OVERVIEW

Computation using formal-language and automata theory. He proved that Adleman's linear self-assembly process [4] is equivalent to generating regular sets. In other words, each linear self-assembly system maps into a finite-state machine and vice versa. Winfree then proposed a new improved self-assembly model in two dimensions called tile self-assembly, a model based on the work of Wang [54] and the theory of Block Cellular Automata (BCA). A tile is a square with encoded edges; tiles can connect to one another if the connecting edges have matching (complementary) encodings. Tiles are fundamental structures of BCA, a Turing universal model of computation.

The question is how to create interacting DNA molecules that can be modeled as Wang tiles and hence perform BCA computations. Winfree et al. [62] proposed that DNA double-crossover molecules originally invented in Seeman’s laboratory could be used as such tiles. The fabrication of DNA double-crossover (DX) molecules was pioneered by Seeman [30] using DNA branched junctions. A DNA DX molecule consists of four "sticky ends" (short strands of DNA) appended to a central complex (Chapter 3 provides an illustration). The sticky ends map into the encoded sides of the tiles; only complementary sticky ends actually bond. These DNA DX molecules self-assemble by bonding at their sticky ends into 1D, 2D, or 3D lattices or crystals. The self-assembly of DNA tiles is controlled by a set of (bonding) rules determined by the sticky-end encodings (nucleotide sequences) of the various DNA tile types. Thus, the rules are local and the tiles that implement those rules are called rule tiles. The work involved in constructing a self-assembly computation essentially comes down to determining the set of tile types, the program of the system. Surveys of recent work in the self-assembly of DNA lattices are given by Reif et al. [46], LaBean et al. [36], and Winfree [60].

Winfree's tile self-assembly model provides the framework for modern theoretical developments in algorithmic self-assembly. Rothemund and Winfree [50] further
studied the complexity issues within the tile self-assembly model using the notion of Kolmogorov complexity. They measured the complexity of tile self-assembly, the program-size complexity, as the minimum number of distinct tiles required to self-assemble an $N \times N$ square; they found that the program-size complexity is $\Theta(\log N)$.

Adleman et al [2] studied both the program-size complexity and time complexity of the problem of self-assembling an $N \times N$ square. Rothemund and Winfree's work in [50] showed that the time complexity of this problem is $\Theta(N \log N)$. Adleman et al showed how this result could be improved by applying certain modifications to the system, e.g., the number of tile types. In particular, they showed that the program-size and time complexity of the $(N \times N)$-square self-assembly problem can be improved to $\Theta(\log N / \log \log N)$ and $\Theta(N)$ at the expense of an additional $\Theta(\log N)$ tile types.

Extending beyond $N \times N$ squares, Adleman et al showed that the program-size complexity of producing an arbitrary shape is NP-hard (see [3]). They also studied the tradeoff between the program size and the computation time in algorithmic self-assembly. For more on the theory of complexity, combinatorial optimization, and algorithmics as it relates to tile self-assembly systems, we refer the reader to the bibliographies in [1, 3, 2, 6, 50, 58].

Chapter 3 below first summarizes the elementary structures of DNA molecules used in computation, and then illustrates how these have been modeled as tiles in computational processes. Specific computations are illustrated as tile sets and tile-bonding sequences leading to crystals whose precise structure contain the results of the computations.

No model of DNA-based computation is complete without an accounting of certain errors that occur in crystal growth. For, such errors occur in practice to an
CHAPTER 1. OVERVIEW

extent that creates a serious error-tolerance design challenge that must be resolved. A number of error-correction and prevention methods have been proposed including notably coding and redundancy techniques for detecting, preventing, or correcting errors (see e.g. [61, 23, 47, 22, 31]). This work is reviewed briefly in Chapter 3, after which our contribution to this problem in Part III is previewed.

The literature on the subjects of this thesis is quite large and still growing fast. As will be seen, the theory of interacting particle systems plays an important role in our studies, and classical references in this area are the books by Liggett [38, 39]. For the literature growing out of the seminal work of Adleman [4, 5, 6], Winfree [56, 59], Rothemund and Winfree [50], and Seeman [30], relevant journals and workshop proceedings include the Proceedings of the Workshop on DNA-based computers, the Proceedings of the Workshops on the Foundations of Nanoscience: Self-Assembled Architectures and Devices, Science, Journal of Computational Biology, the Proceedings of the National Academy of Science, and the DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Tutorial books include DNA Computing: New Computing Paradigms [53] and Theoretical and Experimental DNA Computation [8].
Chapter 2

Models of Self-Assembly Kinetics

Our reference model of self-assembly is cast in the framework of polymerization: the process 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. The process stops when there are no two polymers in the system which can combine. Our 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. Thus, yield is a monotonically, non-decreasing function of time.

Chemical kinetics, in particular reaction rate equations and the chemical networks of Feinberg [27], provides a useful hydrodynamic (fluid) model of molecular self-assembly. In this model, the dynamical system of self-assembly is 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. In our application, rates are normalized and regarded as bonding probabilities, i.e., the reaction rate of a given pair of colliding reactants is the probability that they bond
CHAPTER 2. MODELS OF SELF-ASSEMBLY KINETICS

together.

To help understand the self-assembly process, consider the simplest non-trivial case of self-assembly known as triangle self-assembly. The origin of the term is the factitious example drawn from nanotechnology as illustrated in Figure 2.1. Unit nanorods as shown in the figure can stick to any other at either end. Triangles are the only stable self-assembled structure under the bonding rules of this process. In general,

![Diagram of self-assembly process]

Figure 2.1: Self assembly of nanorods where both ends of the rods are allowed to stick to the ends of other rods. The triangle structure is the only stable self-assembled structure.

when a self-assembly process stops in an absorbing state, the waste consists of the total concentration of stable polymers other than the polymer that defines yield. For the triangle example above, an absorbing state consists only of triangles (trimers) and of pairs (dimers) that can not combine, except in the case when it consists only of trimers and a single monomer. In the continuous hydrodynamic limit of this model, waste is the concentration of the dimers and the yield is the concentration of trimers.

The system of ODEs describing the triangle self-assembly process is given by

\[
\begin{align*}
\frac{dx_1(t)}{dt} &= -2p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \\
\frac{dx_2(t)}{dt} &= p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \\
\frac{dx_3(t)}{dt} &= 2p_{1,2}x_1(t)x_2(t)
\end{align*}
\]  

(2.1)  

(2.2)  

(2.3)
CHAPTER 2. MODELS OF SELF-ASSEMBLY KINETICS

with $x_1(0) = 1$, $x_2(0) = x_3(0) = 0$, $x_1(t), x_2(t)$ and $x_3(t)$ are the respective concentrations of monomers, pairs, and triangles at time $t$, and $p_{i,j}$ is the probability that colliding $i$- and $j$-polymers actually bond. Each ODE in (2.1)–(2.3) describes, for $k = 1, 2, 3$, the rate of change in the concentration of $k$-polymers as a function proportional to the concentrations of the polymers that can participate in bonds creating or eliminating $k$-polymers. If these consist of an $i$-polymer and a $j$-polymer for example, then with the bonding probability $p_{i,j}$ as the constant of proportionality, the rate of change will be proportional to $x_i^2(t)$ if $i = j$, but to $2x_i(t)x_j(t)$, if $i \neq j$, since, given two polymers, there are 2 ways to designate one as an $i$-polymer and the other as a $j$-polymer.

We consider two cases of self-assembly: irreversible and reversible self-assembly. Reversible self assembly includes events where polymers decompose into smaller polymers. For irreversible self-assembly, the yield calculation based on the system of ordinary differential equations reveals unexpected phase transitions; within the space of reaction rates, there is a subspace for which waste vanishes in the fluid model.

In Part II, we study self-assembly models with the equations of chemical kinetics supplying the infrastructure; in particular, yields and phase transitions are computed explicitly for the simpler models and produced from experiments for more general cases. In Chapter II.3, we fully characterize a phase transition for irreversible triangle self-assembly. We observe that a phase transition in this system depends on the relative, rather than absolute, values of the reaction rates. This chapter also shows that, although an explicit solution to the system of ODEs for the evolution of the yield seems out of reach, we can, by a judicious change of variables, still provide formulas for the final yields when the system is in its absorbing state.

Chapter II.4 studies the interesting special case of incremental self-assembly, where
polymers grow one monomer at a time,\(^1\) This chapter also develops a formula for a special case of incremental self-assembly in the limit of large \(N\), where \(N\) is the size of the yield polymer.

Chapter II.5 revisits the triangle self-assembly system but with reversible reactions allowed. Polymers can decompose into two constituent polymers, e.g., a size \(k\) polymer may decompose into two polymers of size \(i\) and \(j\) where \(i + j = k\), and be governed by given reverse-reaction rates. As before, the reversible system can be expressed as a system of first-order, nonlinear ODEs. It is important to note that yields can be much higher in systems with polymer decomposition, as the waste polymers may be able to decompose into polymers which can later recombine with other polymers in a process that leads to higher yields.

In order to achieve 100% yields in minimal time, the forward reaction rates and the reverse reaction rates must be adjusted in such a way that trajectories of the system of ODEs will stay on the eigenvectors once they have been reached. At this time, all polymer concentrations must be such as to allow the system to exhaust all resources when it reaches an absorbing state along the eigenvectors. Chapter II.5 discusses a strategy that achieves 100% yields in minimal time for the triangle self-assembly system.

As already noted, except for special cases, one must rely on numerical methods and experiments to understand self-assembly systems. Solving the continuous systems expressing hydrodynamic limits using numerical methods can be done fairly easily using the current tools we have available such as Mathlab and Octave. However, determining system behavior by experiments (e.g., simulation) requires a greater understanding of the \textit{discrete} self-assembly process underlying the equations of chemical

\(^1\)Note that this generalizes the triangle problem.
kinetics. Chapter II.2 describes the dynamics of an elementary Markov discrete-event chemical process in which polymers moving randomly in a fluid collide or decompose at some given rate. The polymer collision and decomposition processes are taken to be Poisson, and in the hydrodynamic limit, the discrete system becomes the continuous system described by reaction rate equations. Experiments in the discrete world show that, because of the molecular scaling, the analytical results of hydrodynamic limits give extremely good approximations.
Chapter 3

Tile Self-Assembly Models

The seminal mathematical *tile* model of DNA self-assembly, as developed by Winfree [56] and pursued by many others, has led to a much improved understanding of DNA self-assembly in two dimensions. At the physical layer being modeled, single-strand DNA molecules are manipulated to form DNA molecules (e.g., *double-crossover molecules* [26, 30]) which are designed to assemble (bond) with other such molecules in a two-dimensional crystal-growth process obeying bonding rules determined by the molecular motifs [56, 57]. Throughout this thesis, a DNA DX molecule refers to a double-crossover DNA molecule. These building-block molecules are modeled as tiles. Figure 3.1 illustrates an abstract model of a DNA double-crossover molecule which can be used for tile self assembly. The physical model of the DNA structure is shown in Figure 3.1(a). The long lines represent DNA strands and are connected by short vertical lines representing the hydrogen bonds between complementary strands. The strand segments open to bonding are called sticky ends; they bond with the sticky ends of other molecules in the self-assembly process. A cartoon of the physical double-crossover DNA molecule is shown in Figure 3.1(b). The rectangular box in the middle hides the double-crossover structure not needed in a description of self-assembly. Only the four sticky ends are needed for this purpose. The cartoon can be further mapped into a simple tile with edges representing the sticky ends of the
molecule; this is illustrated in Figure 3.1(c).

There are three types of tiles involved in a tile self-assembly process: rule tiles, border tiles, and a seed tile. They participate in a growth process beginning at the origin of the positive lattice; the unit squares of the lattice are the potential sites occupied by tiles.

- The seed tile occupies the lower-left corner of the positive lattice and is respon-
sible for initiating the tile self-assembly process. Only border tiles, as described next, can stick to the two free (upper and right-hand) sides of the seed tiles.

- A border tile can join the structure only by attaching to the seed tile or another border tile along the horizontal and vertical boundaries of the positive lattice, each such attachment extending one of the borders of the structure assembled so far.

- A rule tile can attach to the growing structure at an available site which is adjacent to occupied sites both to the left and below the available site, where either site may contain a border tile.

Winfree's tile self-assembly model requires the temperature\(^\text{1}\) of a self-assembly bond to be two: a tile can attach to another tile or structure if the overall bonding strength of the attachment has a value of at least two. Bonding strength refers to the bonding strength of the hydrogen bonds between two matching sticky ends of two connecting DNA double-crossover molecules. Abstractly, it can be viewed as the strength of the attachment between two matching edges of adjacent tiles.

A border tile can attach to another border tile or to a seed tile with a bonding strength of two. This design allows the border tiles to grow stably in the vertical and horizontal directions, so the initial scaffold of the self-assembled structure can be created as an L-shape; border tiles can serve as the input to a computation. Often shown abstractly throughout our work, a border tile in Figure 3.2 contains thick edges representing a bonding strength of two when matched with other thick edges.

---

\(^1\)In the present context, temperature is a system parameter discretizing and simplifying standard temperature scales in chemical kinetics. It refers here to bonding strength: The temperature needed to break a bond grows with the strength of the bond.
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

A rule tile can attach to the growing structure at any site that is adjacent to occupied sites both to the left and below the site, where either adjacent site may contain a border tile. Figure 3.2 shows an example of tile self-assembly. We assume that the tiles cannot rotate. Border tiles attach to the seed tile in the vertical and horizontal directions as indicated above. In the figure, the edges that provide a bonding strength of one are assigned half-circles in black and white. A black half-circle edge can only attach to a matching black half-circle edge; the same bonding rule applies to white half-circle edges. Two matching half-circle edges form a bond with strength one, but edges of different types cannot bond (have a bonding strength of 0). A rule tile can only attach at a bonding site when the sites to the left and below are already occupied; the left edge of the rule tile matches the right edge of the tile to its left; and its bottom edge matches the top edge of the tile below it.

We have described just one graphical representation of a tile self assembly system. Other variants, more or less equally natural and easily interpreted, can be found here (see, e.g., Figure 3.4) and in the literature.

3.1 Computation by Tile Self Assembly

As discussed in the Section 3, DNA DX molecules can bond with other DNA DX molecules via sticky ends. We are particularly interested in the resulting patterns created by self-assembly processes. To understand how self-assembled patterns of the DNA molecules of Section 3 can be controlled, consider Figure 3.3(a), which illustrates a toy model of DNA tile self-assembly. The lower-left sticky end of the DNA DX molecule of type $D$ bonds with the upper right sticky end of the DNA DX molecule directly under it to the left. The DX molecule's lower-right sticky end also bonds with the upper-left sticky end of the DNA DX molecule directly under and to its right (see the zoom-in image of Figure 3.3(a)). These two bonding edges produce
Figure 3.2: A simple abstract model of two-dimensional tile self-assembly. This process starts with a single seed tile at the origin. The border tiles are allowed to grow in the vertical and horizontal directions away from the seed tile. Rule tiles can only attach to the structure at “corner” sites where there are tiles with matching edges on the left and below the site. There are three types of edges: thick black line, white half-circle and black half-circle. The bonding strength between two thick black lines is two and the bonding strength between two half-circle edges of the same color is one. All other bonds have strength 0.

A bonding strength of two which is enough for stable attachment to the structure. Look upon the new DNA DX molecules as arriving from the top. From the figure, there is a simple attachment rule which can create a well-organized pattern. Since
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

the self-assembly process proceeds in the upward direction, the attachment of each DNA DX molecule can be viewed as the mapping of information from two sticky ends of two different DNA DX molecules in the lower row to the upper sticky ends of the attaching DNA DX molecule.

The zoom-in image of Figure 3.3(a) illustrates the mapping of two inputs $x$ and $y$ of the DNA DX molecules in the lower row to the outputs $f_D(x, y)$ and $g_D(x, y)$ on the upper sticky ends of the attaching DNA DX molecule. The output information encoded in the sticky ends of the DNA DX molecule $D$ will be used as inputs for subsequent tile attachments.

The arrangement of DNA DX molecules in Figure 3.3(a) can be viewed as the tiling of unit squares illustrated in Figure 3.3(b). The tiling of squares relates to the work of Wang [55] who introduced a system of tiles now called Wang tiles. A tile set is a finite set of different tile types. A tile configuration is defined as the tiles in the two-dimensional lattice. A tiling is a tile configuration in which two juxtaposed tiles have the same type on their common border. Wang showed that the simulation of a Turing machine could be achieved with the tiling of a tile set. This relationship between Turing machine and tiling system was formally proved by Lewis and Papadimitriou [37], and they also demonstrated a direct simulation of a single tape deterministic Turing Machine using a tiling system. Figure 3.3 shows how a computation can be performed by tile self-assembly. In the figure, the set of substitution rules $(x, y) \rightarrow (f(x, y), g(x, y))$ is performed by the attachment of tiles usually referred to as rule tiles.

The concept of using DNA DX molecules to simulate Wang tiles was introduced and implemented experimentally by Winfree [56, 62]. The universality of Winfree's tiling system set the stage for many researchers in DNA-based computing. Part III
Figure 3.3: (a) A cartoon of DNA DX molecules self-assembling to form a pattern. The attachment of a DNA DX molecule can be regarded as an operation on the inputs encoded in the sticky ends of DNA DX molecules in the lower row to produce the outputs encoded in the upper sticky ends of the attaching DNA DX molecules. The set of bonding rules determining self assembly can simulate a Turing machine. (b) The pattern shown in (a) can be viewed as a tiling of unit squares.

Examples In 1998, Winfree demonstrated the computational power of DNA-based tiling systems in a construction that solved the Hamiltonian path problem [62]. He
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

designed 68 distinct types of double cross-over DNA molecules to compute solutions to this problem via self assembly. These designs introduce an obvious optimization problem: What is the minimum number of tile types needed to compute a given function? He also showed that computation by linear self-assembly (i.e., self-assembly in one-dimension) is equivalent to the recognition of regular languages and dendrimer self-assembly computation is equivalent to the recognition of context-free languages.

Based on Winfree’s concept, Mao et al [43] designed a set of triple-crossover DNA molecules that can perform the exclusive-or (XOR) operation. Although double-crossover DNA molecules can perform the same computation, the triple-crossover DNA molecule has the advantage of being more rigid, thus more suitable for stable two-dimensional self-assemblies. However, the synthesis of triple-crossover molecules is more difficult.

![Diagram of a parity check computation](image)

Figure 3.4: A parity check computation. In the graphics of this tile model, sticky ends are represented explicitly by symbols at the corners of the tiles. The symbols here are bits corresponding to sticky-end encodings designed for the parity check operation.

Later, in 2002, Carbone and Seeman [21] pointed out that the cumulative XOR computation of Mao et al [43] can also be adapted to a circuit that computes the
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

parity of an input string. Figure 3.4 shows an example of a parity check program using tile self-assembly which uses a different tile graphic. Sticky ends have two different and complementary encodings representing 0 and 1 bits. Bit values label the corners of the various tile types. In the figure, the input string is represented by a sequence of shaded tiles with the bits of the string given by those in the lower-right corners; the remaining corner bits are not shown as they are immaterial. This input is pre-assembled along with two shaded special-purpose input tiles, which are needed to provide the initial rule-tile attachment site for the parity-check self-assembly process. Only rule tiles that have corner bits matching the two input bits at the bonding site are allowed to attach to the input. Each successive attachment of a rule tile creates a new available site allowing another rule tile to attach, until the end of the input-tile sequence is reached. The output of the program, i.e., the parity of the input string, is given by the bit value of the upper-right corner of the last tile attached to the structure.

For another computational primitive of greater complexity, again based on the XOR tiling rules, consider Rothemund’s [51] construction of a binary counter using the seven tile types shown in Figure 3.5. Note that Rothemund’s self-assembly process starts with a structure that consists of pre-assembled border tiles forming an L-shape structure. There are four types of rule tiles corresponding to the XOR truth table. The binding mechanism produces what is in fact a realization of the familiar ripple adder of computer arithmetic units. During the self-assembly process, the \( i \)-th row of tiles represents integer \( i \); the \( j \)-th rule tile from the left gives the \( j \)-th bit in the binary representation of \( i \).

In 2004, Rothemund et al [49] applied algorithmic self assembly to the growth of fractal patterns of Sierpinski Triangles; again the technique is based on the rule tiles that implement XOR local rules, as can be seen in Figure 3.6. As the authors point
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

out, their construction leads to another method for proving the Turing universality of algorithmic self assembly. In this case, they show in effect that algorithmic self assembly can emulate cellular automata, and thus the former inherits the Turing universality of the latter.

3.2 Error-Correction in Tile Self-Assembly

Tile Self-assembly discussed so far is assumed to occur in a perfect world where every tile that attaches to an available site is of the correct type, according to the design of the self-assembly system; and every such tile attaches correctly. But errors in both senses can and do occur in practice and to an extent (error rates up to 10% have been reported) that creates a serious error-tolerance design problem. The tile self-assembly model has very little tolerance for errors in computing applications. An error in a tile attachment corresponds to an error in a computational primitive that can render
all subsequent tile attachments erroneous. Various methods have been used to deal with error-prone computers, including notably coding and redundancy techniques for detecting, preventing, or correcting errors.

The two most frequent errors in DNA self-assembly are growth errors and nucleation errors. Growth errors occur when a wrong type of tile, an error tile, attaches to the lattice; a sublattice that forms with the error tile at its origin will then be corrupt. A nucleation error occurs when only one side of a tile attaches to the lattice, and hence at a wrong position. Thermodynamic controls that slow down growth can be introduced to help ensure the relatively early separation of error tiles.

A tile can also be designed to have its own error-correction capability, or a new type of tile that assists the self-assembly process in lowering error rate can be introduced. Several methods for this have been proposed. For example, Winfree and Bekbolatov's Proofreading Tile Set [61] shows that the error rate can be reduced significantly by
CHAPTER 3. TILE SELF-ASSEMBLY MODELS

creating an original Wang Tile using four or nine smaller tiles (2 × 2 or 3 × 3) in order to ensure that the small incorrect tiles will fall off before they are assembled to form an incorrect Wang tile. Chen and Goel’s Snake Tile Set [23] improves the Proofreading Tile Set by ensuring that the smaller tiles can be assembled only in certain directions.

Reif et al. [47] use pads to perform error checking when a new tile is attached to the lattice. Each pad acts as a kind of adhesive, connecting two Wang tiles together, whereas in the original approach the Wang tiles attach to each other. This method allows for redundancy: a single pad mismatch between a tile and its immediate neighbor forces at least one further pad mismatch between a pair of adjacent tiles. This padding method can be extended further to increase the level of redundancy.

Chen et al’s Invadable Tile Set [22] applies the invading capability of the DNA strand to emulate the invasion of a tile. In this model, the tiles are designed so that the correct tile can invade any incorrect tile during the lattice growth process. Fujibayashi and Murata’s Layered Tile Model [31] significantly reduces the error rate by using two layers of tiles: the Wang tile layer and the protective tile layer. The protective layer does not allow tiles to attach to the lattice incorrectly. When the attachment is correct, the protective tile releases the rule tile, to which the next tile attaches itself. As one must expect, all methods have shortcomings or costs associated with them, such as prolonged self-assembly times, enlarged lattices, potential instabilities, and failure to deal effectively with both error types.

In Chapter III.4, we introduce a checkpointing technique for graceful recovery from errors. This method determines regular checkpoints, or milestones, at which the state of a computation, or the system as a whole, is stored in a reliable memory. When an error occurs the state of the computation can be restored from, i.e., by rolling back to, the state saved at the most recent checkpoint.
In Part III, we move away from the theory of computation as we shift our interest to the engineering aspect of tile self-assembly (e.g. the time requirement to complete an $N \times N$ square). We introduce a stochastic model to describe tile self-assembly systems. In Chapter III.2, we discover a strong relationship between tile self-assembly system and a well-known interacting particle system. Applying results from the latter system, we accurately calculate the time it takes to complete a self-assembly process.
Part II

Self Assembly Models
Chapter 1

Introduction

A useful model of molecular self assembly and polymerization can be adapted from hydrodynamic (fluid) models of chemical kinetics, in particular the chemical network equations of Feinberg [27]. These special cases are justified heuristically in this section. But in Chapter 2 a discrete, molecular process is defined which gives a more concrete justification of these special cases, since it gives the same equations in the fluid limit. The dynamics of the discrete process are simulated in Chapter 2, and for the chosen parameters, behaviors shown by experiments and hydrodynamic limits are in very close agreement.

At a given time $t$, the fluid-limit equations are expressed in terms of concentrations $x_i(t)$ of molecules of “size” $i$, to be called $i$-polymers$^1$ and in terms of given parameters $p_{i,j}$ which specify reaction rates between $i$- and $j$-polymers. The self assembly process has an initial state consisting only of monomers, say $n$ of them. If $\eta_i(t)$ denotes the number of $i$-polymers that have self-assembled at time $t$, then the concentrations are given by the hydrodynamic limits

$$x_i(t) = \lim_{n \to \infty} \frac{\eta_i(nt)}{n}$$  

(1.1)

$^1$Adhering to standard terminology, 1-, 2-, and 3-polymers will be called monomers, dimers, and trimers.
CHAPTER 1. INTRODUCTION

By definition, \( \sum_i i \eta_i(t) = n \), so for all \( t \)

\[
\sum_{i \geq 1} i x_i(t) = 1 \tag{1.2}
\]

and the initial state is now \( x_i(0) = 1 \) if \( i = 1 \) and is 0 for all \( i > 1 \). The dynamical system of self assembly is then a system of first-order, ordinary differential equations (ODEs) giving the rates of change of concentrations in terms of concentrations and reaction rates; these are called self assembly equations in the sequel. Our interest, outside of Chapter 5, is restricted to irreversible self assembly, i.e., to pure growth models of self assembly wherein polymers only bond to form larger polymers; they can not decompose into smaller polymers.

Chapter 2 describes the dynamics of a simple underlying discrete chemical process in which polymers moving randomly in a fluid collide at some given rate; \( p_{i,j} \) is the probability that colliding \( i \)- and \( j \)-polymers actually bond. Behavior in the hydrodynamic limit leads to generalized self assembly equations given later in this section. For the present, to help fix ideas consider the simplest non-trivial case of self assembly, known as triangle self-assembly. The origin of the term is the factitious example drawn from nanotechnology illustrated in Figure 1.1. One imagines the self-assembly of unit nano-rods as shown in the figure, where any nano-rod can stick to any other at either end. Assume that the nano-structures cannot decompose, and triangles are the only stable self-assembled structure under this bonding scheme. The self assembly equations are given by

\[
\begin{align*}
\frac{dx_1(t)}{dt} &= -2p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \\
\frac{dx_2(t)}{dt} &= p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \\
\frac{dx_3(t)}{dt} &= 2p_{1,2}x_1(t)x_2(t)
\end{align*} \tag{1.3, 1.4, 1.5}
\]

with \( x_1(0) = 1 \), \( x_2(0) = x_3(0) = 0 \). Return to the polymer terminology, and note
that, implicitly, $p_{i,j} = 0$ except when $i + j = 2$ or 3; dimers never bond to each other and trimers are inert. By inspection of the signs, the monomer and trimer concentrations are, as required, monotonically non-increasing and non-decreasing functions of time, respectively. The process reaches an absorbing state in which only dimers and trimers can exist, in the sense that the concentrations satisfy $x_1(\infty) = 0$, $x_2(\infty) \geq 0$, and $x_3(\infty) > 0$. The quantity $3x_3(\infty)$ is called the yield of the process and represents the hydrodynamic limit of the fraction of monomers that appear in trimers. The equations are justified as follows.

![Diagram](image)

Figure 1.1: Self assembly of nanorods where both ends of the rods are allowed to stick to the ends of other rods. The triangle structure is the only stable self-assembled structure.

In the discrete, finite-$n$ self assembly process, changes in concentrations are brought about by events in which two polymers bond to form a new, larger polymer. As expected, therefore, the rate of change in the concentration of $k$-polymers is (in the fluid limit) proportional to the concentrations of the polymers that can participate in bonds creating or eliminating $k$-polymers. If these consist of an $i$-polymer and a $j$-polymer for example, then with the bonding probability $p_{i,j}$ as the constant of proportionality, the rate of change will be proportional to $x_i^2(t)$ if $i = j$, but to $2x_i(t)x_j(t)$, if $i \neq j$, since, given two polymers, there are 2 ways to designate one as an $i$-polymer and the other as a $j$-polymer.

For the rate of change of monomers in (1.3), the first term on the right-hand side
corresponds to $i = j = 1$; the factor of 2 reflects the loss of two monomers in this reaction, a role which is to be kept separate from that of the second factor of 2, which arises because the second term on the right-hand side of (1.3) refers to a bond between an $i$-polymer and a $j$-polymer, with $i \neq j$. In this case, $\{i,j\} = \{1,2\}$ and a single monomer is lost in bonding with a dimer. The first reaction in (1.3) is also represented in the first term on the RHS of (1.4), where a dimer is created by the bonding of two monomers. The second terms on the right of (1.3) and (1.4) are the same and represent the loss of a monomer in the first case, and a dimer in the second case, when a dimer bonds with a monomer to form a trimer. With a change in sign, this term reappears in (1.5) as the single, trimer-concentration growth term. Clearly, all possible ways of eliminating monomers, eliminating or creating dimers, and creating trimers are represented in the fluid limit of triangle self assembly described by (1.3)–(1.5).

Since the sum of the $ix_i(t)$ is 1, only solutions for $x_1(t)$ and $x_2(t)$ from (1.3) and (1.4) are needed. As noted, the monomer concentration in the absorbing state is 0, so the yield is given by

$$\beta := 3x_3(\infty) = 1 - 2x_4(\infty)$$  \hspace{1cm} (1.6)

Correspondingly, the waste is the dimer concentration in the absorbing state. As discussed at length in Chapter 3, there is a subset of the parameter space where the waste is 0.

The triangle case is easily extended to the case where $N$-polymers, $N > 3$, constitute the yield: $N$-polymers are inert and for all $k > N$, $k$-polymers can not form, i.e., $p_{i,j} = 0$ for all $i,j$ with $i + j > N$. One obtains, suppressing the dependence on $t$, 

\[
\frac{dx_1}{dt} = -2x_1 \sum_{k=1}^{N-1} p_{1,k} x_k \\
\frac{dx_k}{dt} = -2x_k \sum_{j=1}^{N-k} p_{j,k} x_j + \sum_{\{i,j:i+j=k\}} (1 + 1_{i\neq j}) p_{i,j} x_i x_j
\] (1.7) (1.8)

with \(x_1(0) = 1\) and \(x_i(0) = 0, \ i > 1\). In (1.7), the first term of the sum reflects the loss of two monomers while the remaining terms reflect the loss of just one. A similar explanation in terms of \(k\)-polymers applies to the first sum in (1.8). The second sum expresses the growth of \(k\)-polymers when \(i\)- and \(j\)-polymers bond, where \(i + j = k\) (the coefficients are all 2 except when \(i = j\)).

Throughout this work, the reaction rates \(p_{i,j}\) are taken to be symmetric, so hereafter \(p_{i,j}\) is defined only for \(1 \leq i \leq j\).

Equations (1.7) and (1.8) comprise a system of first-order nonlinear ODEs for which one can not expect explicit solutions in cases of practical interest, even when \(N = 3\). Remarkably, however, one can solve for the yields of the self assembly process in interesting special cases without actually finding the functions \(x_i(t)\). This will be the subject of chapters 3 and 4.

So far, we have said nothing about the times between bonding events in the underlying discrete self assembly process. Chapter 2 studies this issue when simulations are used to estimate the functions \(x_i(t)\). For purposes of computing yields, however, the distribution of inter-event intervals is not needed. We may think of a trajectory of the self assembly process as simply a sequence of self assembly events. Normally, parameters are chosen so that these sequences are finite with probability 1 and end with the system in an absorbing state.
CHAPTER 1. INTRODUCTION

Chapter 2 presents a simulation tool and shows that, in the appropriate limit, the equations underlying the discrete-event processes being simulated tend to the reaction rate equations. In the two chapters to follow, we cover those cases where analysis of the reaction rate equations has led to explicit results for yields. Chapter 3 begins with the special case where the yield parameter is infinite; the concentrations as a function of time are calculable in this case. Then the simplest nontrivial case for finite yields is analyzed, viz. the case $N = 3$ of triangle self assembly. This analysis leads to studies of unexpected phase transitions where, below a certain threshold on $p_{1,1}$, with the remaining reaction rates held fixed, 100% yields always occur, i.e., $x_1(\infty) = x_2(\infty) = 0$ and $\beta = 1$.

Chapter 4 then studies yields in an interesting special case of incremental self assembly, where polymers grow one monomer at a time. Precisely, for a given yield parameter $N > 2$, $p_{i,j} > 0$ if $i = 1$, $i \leq j \leq N - 1$; otherwise, $p_{i,j} = 0$. In the special case studied, $p_{i,j} = 1$ if $i = j$ and $= 1/2$ if $i \neq j$. Finally, Chapter 5 discusses reversible self assembly where polymers can decompose into smaller polymers as well as bond to form larger polymers.
Chapter 2

An Experimental Tool

As noted earlier, with explicit solutions in mind, the reaction rate equations of chemical kinetics have rarely been tractable; even numerical approaches have been limited within the parameter spaces of practical systems. Moreover, when large cluster/polymer sizes are possible, even standard mathematical software tools are unsatisfactory. Thus, studies of algorithmic self-assembly have turned to experimental tools based on discrete-event simulation. This chapter presents such a model argued from first principles and elementary collision theory. We prove first that in the hydrodynamic limit the equations underlying the discrete model become reaction rate equations of Feinberg's chemical networks [27]; and second, we illustrate with specific experiments that the accuracy of our experimental results is strikingly good, even for systems that are very small, i.e., with relatively small populations of molecules.

For purposes of illustration, we study incremental self assembly keeping with polymerization terminology: $p_{ij} = 0$ for all $i > 1$ so that a monomer participates in every reaction with probability 1. With this condition applied to (1.7) and (1.8), the reaction rate equations in the irreversible case are given below, with the simpler notation...
\[ p_j := p_{1,j}, \]

\[ \frac{dx_i(t)}{dt} = 2x_1(t)x_{i-1}(t)p_{i-1} - 2x_1(t)x_i(t)p_i, \quad i > 2 \]

\[ = x_1^2(t)p_1 - 2x_1(t)x_2(t)p_2, \quad i = 2 \]

\[ = -2 \sum_{l \geq 1} p_l x_1(t)x_l(t), \quad i = 1 \tag{2.1} \]

The approach of Markovian birth-and-death equations to chemical kinetics is well studied [32], although it is often justified by its more realistic modeling of random molecular collision/bonding phenomena. Here, our interest in simulation is confined to its use as a means of studying systems too large for other techniques, e.g., the numerical solution of systems of ODEs; in particular, we are interested in scalings where reaction rate equations are very accurate models, but the number of equations is so large that exact results are computationally infeasible. Our derivation of the polymerization equations is notable for its simplicity. And our results include useful estimates of convergence to the reaction rate equations of the hydrodynamic limit. These estimates show that the simulation of a relatively, very small number of molecules (many orders of magnitude smaller than the number in the systems being modeled) is surprisingly close to the hydrodynamic limit.

In our discrete Markov collision-event model of self assembly, times between successive pair-wise collisions of polymers are independent with a time-dependent rate parameter \( R(t) \) which reflects reactant concentrations at time \( t \). Consistent with elementary collision theory [25], we take \( R(t) \) as proportional to the concentrations of reactant polymers in a host fluid, i.e., \( R(t) = \lambda \eta^2(t) \), where the constant, \( \lambda \), aggregates various underlying physical parameters affecting the collision rates between polymer pairs, and where \( \eta(t) \) is the total number (the unnormalized concentration) of polymers in the system at time \( t \). The parameter \( p_j \) is the probability that a given collision between a monomer and a \( j \) polymer is effective, i.e., results in a reaction that produces a new polymer.
Thus, a trajectory of irreversible incremental self assembly is created by a sequence of independent samples from the exponential distribution with parameter $R(t_k)$, where $t_k$, $k = 1, 2, \ldots$ is the time of the $k$-th two-polymer collision. At each such collision, the colliding polymers are chosen independently from the distribution $\{\eta_1(t)/\eta(t), \ldots, \eta_N(t)/\eta(t)\}$. The two polymers, say $i$- and $j$-polymers bond only if at least one is a monomer, and then only with probability $p_j$ or $p_i$, depending on which is a monomer. Note that a maximum polymer size, say $N$, is implicit in the model by stipulating that $p_j = 0$ for all $j \geq N$.

To develop an ODE for $\eta_i(t)$, consider the evolution of the number of $i$-polymers in the interval $[t, t + \Delta t]$ given that a single collision occurs. At time $t + \Delta t$ in incremental self assembly, the number of $i$-polymers will increase by 1 if the collision is between a monomer and an $(i-1)$-polymer, and they bond to produce an $i$-polymer; the number decreases by 1 if the collision is between a monomer and an $i$-polymer, and they bond to produce an $(i+1)$-polymer; and the number stays the same otherwise.

Draw two samples from the polymer population to determine a collision sample; if, say, the first sample is an $i$-polymer and the second a $j$-polymer, then the sample is denoted by $(i, j)$. For $i \neq j$, the samples $(i, j)$ and $(j, i)$ are identical in effect, i.e., they constitute the same collision, so the probability of such a collision is simply twice the probability of either ordered sample $(i, j)$ or $(j, i)$ alone. For simplicity, we denote collisions by $\{i, j\}$, with $i \leq j$, and their probabilities by

$$\Pr\{i, j\} = \begin{cases} 2 \frac{\eta_i(t) \eta_j(t)}{\eta(t)} & i \neq j, \\ \left(\frac{\eta_i(t)}{\eta(t)}\right)^2 & i = j \end{cases}$$

Now let $\Psi_i$, $i > 1$, denote the set of all polymer collisions except those involving a monomer and either an $i$-polymer or an $(i-1)$-polymer. Then for $i > 1$ with
\( \eta_i(t) > 0 \), we have in the usual way, by the properties of the collision process,

\[
\eta_i(t + \Delta t) = \eta_i(t)(1 - R(t)\Delta t) + (\eta_i(t) + 1)R(t)\Delta t \Pr\{1, i - 1\}p_{i-1} \\
+ \eta_i(t)R(t)\Delta t \sum_{\{k, l\} \in \psi_i} \Pr\{k, l\} + \Pr\{i - 1\}(1 - p_{i-1}) \\
+ \Pr\{1, i\}(1 - p_i) + (\eta_i(t) - 1)R(t)\Delta t \Pr\{1, i\}p_i
\]

Applying \( \sum_{\{k, l\}} \Pr\{k, l\} = 1 \), ignoring terms on the order of \( o(\Delta t) \), and substituting for \( R(t) = \lambda \eta(t)^2 \) and the conditional collision probabilities, we rearrange to obtain, in the limit \( \Delta t \to 0 \), the desired incremental polymerization equations for \( \eta_i(t) \).

\[
\frac{d\eta_i(t)}{dt} = \lambda (2\eta_i(t)\eta_{i-1}(t)p_{i-1} - 2\eta_i(t)\eta_i(t)p_i) \quad i > 2 \tag{2.2}
\]

and for \( i = 2 \)

\[
\frac{d\eta_2(t)}{dt} = \lambda (\eta_1^2(t)p_1 - 2\eta_1(t)\eta_2(t)p_2) \tag{2.3}
\]

After introducing the convenient normalization \( \lambda = 1 \) and then the LLN scaling \( x_i^n(t) := \frac{n(t_i^n)}{n} \), where \( n \) is the initial number of monomers in the system, we then take the limit \( x_i(t) := \lim_{n \to \infty} x_i^n(t) \) to obtain for \( i \geq 2 \), as claimed, the reaction rate equations given earlier in (2.1).

The monomer case \( i = 1 \) remains, and it is quite different: First, monomers can bond with all \( i \)-polymers, \( 1 \leq i \leq N - 1 \), and second, they can not be created by bonds; \( \eta_1(t) \) is monotone nonincreasing in \( t \). Given that two polymers collide and that \( \eta_1(t) \geq 2 \), we have \( \eta_1(t + \Delta t) = \eta_1(t) \) if neither is a monomer, or if one is a monomer but no bonding occurs, \( \eta_1(t + \Delta t) = \eta_1(t) - 1 \) if one is a monomer and the other is a \( j \)-polymer for some \( j \geq 2 \) and bonding occurs, or \( \eta_1(t + \Delta t) = \eta_1(t) - 2 \) if both are monomers and they bond. Then,
\[ \eta_i(t + \Delta t) = \eta_i(t)(1 - R(t)\Delta t) + \eta_i(t)R(t)\Delta t \left( \sum_{(k,l) \in \Psi_i} \Pr\{k,l\} + \sum_{(1,l), l \geq 1} (1 - p_l) \Pr\{1,l\} \right) + [\eta_i(t) - 1]R(t)\Delta t \sum_{(1,l), l \geq 2} p_l \Pr\{1,l\} + [\eta_i(t) - 2]R(t)\Delta t p_1 \Pr\{1,1\} \]

and so, after manipulations as above, we get

\[ \frac{d\eta_i(t)}{dt} = -2 \sum_{(1,l), l \geq 2} p_l \eta_i(t) \eta_i(t) - 2p_1 \eta_i^2(t) = -2 \sum_{(1,l), l \geq 1} p_l \eta_i(t) \eta_i(t) \]  

(2.4)

Except for the trivial cases where \( \eta_i(t) \leq 1 \), or \( \eta_i(t) = 0 \), \( i > 0 \), which we leave to the reader, we have exhibited the counts \( \eta_i(t) \) as the solution to the nonlinear, first-order ODEs in (2.1), as claimed.

**Accuracy**

To get an idea of the accuracy of the experimental approach, consider the case where the initial number of monomers, \( \eta(0) \), is many orders of magnitude smaller than the number of elementary particles in the laboratory experiments of molecular self assembly being modeled. Figure 2.1 compares the experimental results estimating the function \( \eta_2(t), \ t \geq 0 \), for the time intervals shown, with those obtained numerically from solutions to the corresponding systems of ODEs. As can be seen, the scaling of the figures shown here makes it visually difficult to distinguish between the exact and experimental results.
Figure 2.1: Plots for the simulation and numerical results for $\eta_2(t)$ with $N = 3, 4, 5, 6$. The bonding probabilities are given by $p_j = 0.5$, $1 < j < N$; $p_1 = 0.4, 0.1, 0.05, 0.04$ for $N = 3, 4, 5, 6$. We start these experiments with $\eta(0) = 50000$ monomers for $N = 3, 4$ and $\eta(0) = 100000$ for $N = 5, 6$.

These results beg the question of just how small $\eta(0)$ can be and still have highly accurate results. A partial answer to this question was obtained by investigating the difference (given as a percentage of $\eta(0)$) in yields obtained from experimental and numerical results for the parameters shown in the tables below, where $\eta(0)$ ranges from 500 to 30,000. These data suggest the remarkable fact that $\eta(0)$ values as low as 10,000 yield excellent accuracy throughout.
### Chapter 2. An Experimental Tool

<table>
<thead>
<tr>
<th>$p_1/\eta(0)$</th>
<th>$N = 3, p_2 = 0.5$</th>
<th>$N = 4, p_3 = p_2 = 0.5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.25</td>
<td>0.5</td>
</tr>
<tr>
<td>500</td>
<td>0.922%</td>
<td>0.702%</td>
</tr>
<tr>
<td>1000</td>
<td>0.842%</td>
<td>2.182%</td>
</tr>
<tr>
<td>8000</td>
<td>0.267%</td>
<td>0.272%</td>
</tr>
<tr>
<td>30000</td>
<td>0.024%</td>
<td>0.041%</td>
</tr>
</tbody>
</table>
Chapter 3

Analytical Insights

As often happens in nature, large polymers precipitate, so the case when the polymer size has no bound is of interest as an approximation for the case when $N$ is very large. The $N = \infty$ analysis with unit reaction rates leads to explicit results for time dependent behavior, although the notions of yield and absorbing state clearly degenerate. This chapter covers this case in the next section, and will then move on, in the subsequent section, to the simplest nontrivial case of finite $N$, viz., the case $N = 3$ of triangle self assembly. In this case, we are unable to solve for the concentrations as functions of time, but we are able to compute yields, the fraction of the original number of monomers that appear in the trimers of the absorbing state. We also discover phase transitions whereby, beyond a certain threshold on parameter values, no waste occurs. The chapter concludes with experimental studies of systems not amenable to analysis leading to explicit results.

3.1 Self Assembly with $N = \infty$.

From (1.7) and (1.8) and the assumed unit reaction rates $p_{i,j}$, we obtain, for $\ell \geq 1$,

$$\frac{dx_\ell}{dt} = -\sum_{i=1}^{\infty} 2x_i x_\ell + \sum_{m+n=\ell} x_m x_n$$
CHAPTER 3. ANALYTICAL INSIGHTS

To put this in terms of the generating function \( f(z,t) = \sum_{\ell=1}^{\infty} x_\ell(t) z^\ell \), multiply by \( z^\ell \) and sum to obtain

\[
\sum_{\ell=1}^{\infty} \frac{dx_\ell}{dt} z^\ell = -2 \sum_{\ell=1}^{\infty} \sum_{i=1}^{\infty} x_i x_\ell z^\ell + \sum_{\ell=1}^{\infty} \sum_{m+n=\ell} x_m x_n z^\ell
\]

\[
= -2 \sum_{i=1}^{\infty} x_i \sum_{\ell=1}^{\infty} x_\ell z^\ell + \sum_{m=1}^{\infty} x_m z^m \sum_{\ell=m}^{\infty} x_{\ell-m} z^{\ell-m}
\]

The product of the sums in the first term can be written as \( f(1,t) f(z,t) \) and the second term can be recognized as the generating function of the sequence \( x_1, x_2, \ldots \) convolved with itself, and hence is equal to \( f^2(z,t) \):

\[
\frac{\partial f(z,t)}{\partial t} = -2f(z,t)f(1,t) + f^2(z,t).
\]

Now set \( z = 1 \) and solve for \( \phi(t) \equiv f(1,t) \) to obtain

\[
\frac{d\phi}{dt} = -2\phi(t)\phi(t) + \phi^2(t) = -\phi^2(t)
\]

and hence, applying the initial condition \( \phi(0) = 1 \) (since \( x_1(0) = 1, x_i(0) = 0, i > 1 \)), an integration gives

\[
\phi(t) = \frac{1}{t+1}
\]

whereupon,

\[
\frac{\partial f(z,t)}{\partial t} = -2f(z,t)f(1,t) + f^2(z,t)
\]

\[
= -2\frac{f(z,t)}{t+1} + f^2(z,t)
\]

(3.1)

Now fix \( z \), compact the notation by defining \( y(t) := f(z,t) \), and rewrite the above equation as

\[
\frac{dy}{dt} + \frac{2}{t+1} y = y^2
\]

(3.2)
with the boundary condition \( y(0) = f(z, 0) = z \) (again, since initially the concentration of monomers is 1 and all other concentrations are 0).

This an instance of the Bernoulli equation, the general form of which has \( y^n \) \((n > -1)\) in place of \( y^2 \). The trick to solving this nonlinear equation is to change variables \( v = 1/y^{n-1} \) which in our case for \( n = 2 \) converts (3.2) to the linear ODE

\[
\frac{dv}{dt} - \left( \frac{2}{t+1} \right) v = -1
\]  

(3.3)

The standard method of integrating factors (in this case, the function \( 1/(1+x)^2 \))
yields the solution \( v = (t + 1) + C(t + 1)^2 \) and hence

\[
y = f(z, t) = \frac{1}{(t + 1) + C(t + 1)^2}
\]  

(3.4)

where \( C \) is a constant of integration. The boundary condition \( y(0) = f(z, 0) = z \)
implies \( C = (z-1)/z \) so that, after rearranging terms,

\[
f(z, t) = \frac{z}{(t + 1)^2} \left\{ \frac{\frac{1}{1 - \frac{t}{t+1} z}}{1} \right\}
\]  

(3.5)

Expanding the bracketed term in a power series gives

\[
f(z, t) = \frac{z}{(t + 1)^2} \sum_{k=0}^{\infty} \left( \frac{t}{t + 1} \right)^k z^k
\]

\[
= \frac{1}{(t + 1)^2} \sum_{k=1}^{\infty} \left( \frac{t}{t + 1} \right)^{k-1} z^k
\]  

(3.6)

and thus the solution to our original problem

\[
x_k(t) = \frac{1}{(t + 1)^2} \left( \frac{t}{t + 1} \right)^{k-1}, \quad k \geq 1
\]

Thus, the polymer concentrations decay geometrically fast in the polymer size at a rate that decreases as time increases. For small \( t \) the concentrations drop precipitously to negligible values, whereas for large \( t \), the concentrations are all small and
spread further and further out to large polymer sizes.

One can easily adjust this method of solution to the situation where the rate matrix \( \{p_{i,j}\} \) is of rank 1, i.e., when

\[
p_{i,j} = p_i p_j
\]

for some sequence \( \{p_i\}_{i \geq 1}, p_i \geq 0 \), so that we acquire a separability \( p_{i,j} x_i x_j = (p_i x_i)(p_j x_j) \) that one can exploit in an obvious way.

### 3.2 Triangle Self Assembly

Our focus in what follows is calculations of yield, leaving the much more difficult problem of time dependent behavior \( x_1(t), x_2(t), x_3(t) \) to experimental studies. Initially, a yield analysis is presented for the triangular self-assembly system formalized in Chapter II.1. This analysis leads to intriguing phase transitions which are then explained analytically. Phase transitions for larger \( N \) are investigated experimentally.

Recall the first two equations for the triangular case (cf. (1.3)–(1.5))

\[
\frac{dx_1(t)}{dt} = -2p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \tag{3.7}
\]
\[
\frac{dx_2(t)}{dt} = p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) \tag{3.8}
\]

with \( x_1(0) = 1, \ x_2(0) = 0 \), where the yield is given by

\[
\beta = 3x_3(\infty) = 1 - x_1(\infty) - 2x_2(\infty) = 1 - 2x_2(\infty)
\]

A key observation at this point is that \( x_1(t) \) is a factor of each term on the right-hand sides of (3.7) and (3.8), so we can convert the system to a \textit{linear} system by introducing a new variable \( s \), with \( ds = x_1(t)dt \). Note that \( s(t) \) is an increasing
function of $t$ with $s(0) = 0$. By inspection, substituting $y_i(s) := x_i(t(s))$, we get

\[
\begin{align*}
  dy_1(s) &= [-2p_{1,1}y_1(s) - 2p_{1,2}y_2(s)]ds \\
  dy_2(s) &= [p_{1,1}y_1(s) - 2p_{1,2}y_2(s)]ds,
\end{align*}
\]  

(3.9)

a system solvable in the standard way in terms of the eigenvalues and eigenvectors of the coefficient matrix. However, we can avoid this work by noting that the system also yields a single differential equation for $y(x) = y_2(x)$, $x = y_1$,

\[
\frac{dy}{dx} = \frac{y - (\frac{\rho}{2})x}{y + \rho x}
\]  

(3.10)

where the two reaction rates have reduced to the single parameter $\rho = p_{1,1}/p_{1,2}$, and where $y(1) = 0$ (since the initial condition is $x_1(0) = y_1(0) = 1$ and $x_2(0) = y_2(0) = 0$). The form of this equation suggests that we divide through by $x$, let $z = z(x) = y(x)/x$, and write

\[
\frac{dz}{dx} + \frac{z}{x} = \frac{z - (\frac{\rho}{2})}{z + \rho}
\]  

(3.11)

and again $z(1) = 0$. Rearranging and then integrating gives

\[
- \int \frac{z + \rho}{z^2 + (\rho - 1)z + \frac{\rho^2}{2}} dz = \log x + C
\]  

(3.12)

with $c$ a constant of integration to be determined. (The symbol $c$ will be used generically to denote constants in what follows.) In terms of the roots $z_1, z_2$ of the quadratic in (3.12), express the integrand as $(z + \rho)/(z - z_1)(z - z_2)$, and expand into partial fractions to get

\[
- \int \left( \frac{A}{z - z_1} + \frac{B}{z - z_2} \right) dz = \log x + c
\]  

(3.13)

where

\[
A = \frac{z_1 + \rho}{z_1 - z_2}, \quad B = 1 - A = -\frac{z_2 + \rho}{z_1 - z_2}
\]  

(3.14)
CHAPTER 3. ANALYTICAL INSIGHTS

with

\[ z_1 = \frac{1 - \rho - \sqrt{\rho^2 - 4\rho + 1}}{2} \]
\[ z_2 = \frac{1 - \rho + \sqrt{\rho^2 - 4\rho + 1}}{2} \]  \hspace{1cm} (3.15)

and

\[ z_1 + z_2 = 1 - \rho \]
\[ z_1 z_2 = \frac{\rho}{2} \]

An integration then yields

\[ c x = \frac{1}{(z - z_1)^A(z - z_2)^B} \]

with the condition \( z(1) = 0 \) requiring that

\[ c = (-z_1)^{-A}(-z_2)^{-B} \]

which yields,

\[ x = \frac{1}{(1 - z/z_1)^A(1 - z/z_2)^B} \]  \hspace{1cm} (3.16)

and

\[ y = z x = \frac{z}{(1 - z/z_1)^A(1 - z/z_2)^B} \]
\[ = \left( \frac{1}{z - z_1} \right)^{-A} \left( \frac{1}{z - z_2} \right)^{-B} \]  \hspace{1cm} (3.17)

with \( A \) and \( B \) given by (3.14).

The objective now is to find an expression for \( y(\infty) = x_2(\infty) \), the limiting concentration of dimers; \( 2y(\infty) \) will then be the waste and \( 1 - 2y(\infty) \) will be the yield. The desired expression takes different forms depending on the value of \( \rho \), the ratio of the reaction rate \( p_{1,1} \) of two colliding monomers to the reaction rate \( p_{1,2} = p_{2,1} \) of a colliding monomer and a dimer. The limiting concentration varies with \( \rho \) as follows.
1. If $\rho \geq 2 + \sqrt{3}$, then $z_1$ and $z_2$ are both real and negative since

$$z_1 = \frac{1 - \rho - \sqrt{\rho^2 - 4\rho + 1}}{2} < -\frac{1 + \sqrt{3}}{2}$$

and

$$z_2 = \frac{1 - \rho + \sqrt{\rho^2 - 4\rho + 1}}{2} < \frac{1 - \rho + (\rho - 2)}{2} = -\frac{1}{2}$$

In particular, as $\rho \to \infty$,

$$z_1 \sim -\rho, \quad z_2 \to -\frac{1}{2}$$

Then from (3.17) and $x(\infty) \to 0$, we have

$$y(\infty) := \lim_{t \to \infty} x_2(t) = \left( -\frac{1}{z_1} \right)^A \left( -\frac{1}{z_2} \right)^B$$

$$= (-z_1)^{\frac{1-z_2}{z_1-z_2}} (-z_2)^{\frac{1-z_1}{z_2-z_1}}$$

with $A$ and $B$ given by (3.14).

2. If $2 - \sqrt{3} < \rho < 2 + \sqrt{3}$, then $\rho^2 - 4\rho + 1 < 0$ and so the roots $z_1, z_2$ are complex conjugates. We represent $z_1 = \bar{z}_2$ in polar form as $|z|e^{i\theta}$ and observe that, since $z_1 z_2 = |z|^2 = \frac{\rho}{2}$ and $z_1 + z_2 = 1 - \rho$, we have

$$(-z_1)^A (-z_2)^B = |z|e^{i(A-B)(\theta-\pi)}$$

with $A$ and $B$ given by

$$A = \frac{1}{2} \left( 1 - \frac{1 + \rho}{\sqrt{\rho^2 - 4\rho + 1}} \right)$$

$$B = \frac{1}{2} \left( 1 + \frac{1 + \rho}{\sqrt{\rho^2 - 4\rho + 1}} \right)$$
and

\[ |z| = \sqrt{\frac{\rho}{2}} \]
\[ \theta = \arccos \left( \frac{1 - \rho}{\sqrt{2\rho}} \right) \]

Then

\[ y(\infty) = \sqrt{\frac{\rho}{2}} \exp \left[ \left( \frac{1 + \rho}{\sqrt{-\rho^2 + 4\rho - 1}} \left( \arccos \left( \frac{1 - \rho}{\sqrt{2\rho}} \right) - \pi \right) \right) \right] \] (3.18)

3. The most interesting behavior occurs when \( 0 < \rho \leq 2 - \sqrt{3} \); the roots \( z_1 \) and \( z_2 \) are both real and negative, and \( z_1, z_2 > -1 \). The concentration of dimers (along with the concentration of monomers) in the absorbing state converges to zero; there is no waste and all polymers are trimers. In particular, \(|z_2|\) converges to zero as \( \rho \) approaches zero.

In other words, this dynamical system implies a phase transition in the waste \( x_2(\infty) \); the waste goes from zero (for all \( \rho < \rho_* = 2 - \sqrt{3} \)) to positive values (for all \( \rho > \rho_* = 2 - \sqrt{3} \)).

Experiments and further insights on phase transitions are covered in Chapter II.2, but a purely analytical exegesis is as follows. The system of linear ODEs

\[ \frac{d\hat{y}_1(s)}{ds} = -\rho \hat{y}_1(s) - \hat{y}_2(s) \]
\[ \frac{d\hat{y}_2(s)}{ds} = \frac{\rho}{2} \hat{y}_1(s) - \hat{y}_2(s) \]

yields (3.10) and can be solved directly. The eigenvalues solve the characteristic equation \((\rho + \lambda)(1 + \lambda) + \rho/2 = 0\) and are given by

\[ \lambda_{1,2} = \frac{-(\rho + 1) \pm \sqrt{\rho^2 - 4\rho + 1}}{2} \]
CHAPTER 3. ANALYTICAL INSIGHTS

The trajectories of the system, as they follow the directions of the eigenvectors, help explain the phase transition. When \( \rho \geq 2 + \sqrt{3} \), the linear system describing the trajectories has negative real eigenvalues, and as \( s \) increases the eigenvector directions show that trajectories enter the second and fourth quadrants where the approach to \((0,0)\) from any initial state is infeasible (in a solution \( \hat{y}_1 \) and \( \hat{y}_2 \) are both positive); stable equilibrium points have only \( \hat{y}_1(\infty) = 0 \). The general behavior is sketched in Figure 3.1. When \( 2 - \sqrt{3} < \rho < 2 + \sqrt{3} \), the eigenvalues become complex and similar observations apply to spiraling trajectories. When \( 0 < \rho < 2 - \sqrt{3} \), the eigenvalues become positive and real. The eigenvector directions show that, as \( s \) increases, the trajectories enter the first quadrant where the approach to the stable equilibrium point at the origin \((0,0)\) is feasible. Thus, the bifurcation that takes place at \( 2 - \sqrt{3} \) corresponds to a phase transition, where \( \hat{y}_2(\infty) \) remains at 0 for all smaller values of \( \rho > 0 \) and remains at positive values for all larger values of \( \rho \).

![Figure 3.1: The trajectories of the system of self-assembly of trimers. The x-axis represents the concentration of monomers, and the y-axis represents the concentration of dimers. (a) \( 0 < \rho < 2 - \sqrt{3} \), the eigenvalues become positive and real. (b) \( 2 - \sqrt{3} < \rho < 2 + \sqrt{3} \), the eigenvalues become complex. (c) \( \rho \geq 2 + \sqrt{3} \), the eigenvalues become negative and real.](image)

Experiments

We have determined yields in the absorbing state, but we have been unable to obtain explicit results for the time-dependent concentrations \( x_i(t) \), even for the relatively
CHAPTER 3. ANALYTICAL INSIGHTS

simple triangle case. For this reason, we turn to experimental tools to give us further insights into system behavior. In this section, we discuss behavior obtained via experiments, as discussed in Chapter II.2. In particular, we focus on the evolution of polymer concentrations, and on a broader characterization of phase transitions.

For a simple system such as the self-assembly of triangles, we can obtain the concentrations $x_i(t), 1 \leq i \leq 3$, using mathematical software tools for solving systems of ODEs, or, for systems with especially large yield parameters $N$, using discrete event simulations. Figure 3.2 illustrates $x_i(t)$ for the triangle system with two different pairs of $p_{1,1}$ and $p_{1,2}$.

The unimodal shape of $x_2(t)$ in Figure 3.2 can be verified analytically from (1.3)–(1.5). The location of the peak can also be computed from these equations by setting the derivative of $x_2(t)$ to 0. This gives

$$p_{1,1}x_1^2(t) - 2p_{1,2}x_1(t)x_2(t) = 0$$

and so the peak is reached at that time $t$ for which $p_{1,1}x_1(t) = p_{1,2}x_2(t)$. This relationship is confirmed in Figure 3.2. In Figure 3.2(a), $x_2(t)$ is maximum when $x_1(t) = x_2(t)$ since $p_{1,1} = 2p_{1,2}$, and in Figure 3.2(b), $x_2(t)$ is maximum when $x_1(t) = 2x_2(t)$ since $p_{1,1} = p_{1,2}$.

We can also investigate the components $x_1(t), 2x_2(t), 3x_3(t)$ of the total (unit) mass as they evolve. Figure 3.3 shows this behavior, where the region between the $x_1(t)$ curve and $x_1(t) + 2x_2(t)$ curve represents the mass of dimers, and the region between $x_1(t) + 2x_2(t)$ and 1 represents the mass of trimers.

Figure 3.4 illustrates a phase transition for the self-assembly of triangles. In the figure, the waste polymers (dimers) given in terms of fluid-limit concentrations is plotted with the corresponding ratio $\rho = p_{1,1}/p_{1,2}$. For $\rho \leq 2 - \sqrt{3}$, the system leaves
no waste in its absorbing state, and the amount of waste is an increasing function of $\rho$ for $\rho > 2 - \sqrt{3}$.

For self-assembly systems with $N > 3$, we have no explicit result for the waste in absorbing states. We rely on numerical solutions and experiments (the latter especially when $N$ is very large) for insights into phase transitions. Figure 3.5 illustrates phase transitions in self-assembly systems with $N = 4$ for various values of the probability parameters $p_{i,j}$. The data for the figure are obtained experimentally with
Figure 3.3: Preservation of mass for the self-assembly of triangles.

Figure 3.4: The value of $\rho = p_{1,1}/p_{1,2}$ determines the phase transition for the self-assembly of triangles.

$\eta(0) = 20000$. The figure suggests the conjecture that phase transitions exist for all values of $p_{1,2}$ and $p_{1,3}$, if $p_{1,1}$ is taken sufficiently small.

We can also gain valuable insights by looking at the plots in Figure 3.5. Figures 3.5(a), (b), and (c) show that, for $p_{1,2} = p_{1,3}$, the leftover dimers are more numerous.
than the leftover trimers in the absorbing state for high values of $p_{1,1}$, since the generation rate of dimers is higher than their decay rate. This effect is amplified when $p_{1,2}$ is smaller than $p_{1,3}$, as seen in Figure 3.5(d). In Figure 3.5(e), the leftover trimers are more numerous than the leftover dimers, because the dimers combine with monomers very quickly in producing large numbers of trimers.
Figure 3.5: Phase transition for the self-assembly system with $N = 4$. 

(a) $p_{1,2} = p_{1,3} = 0.25$

(b) $p_{1,2} = p_{1,3} = 0.5$

(c) $p_{1,2} = p_{1,3} = 0.75$

(d) $p_{1,2} = 0.25, p_{1,3} = 0.75$

(e) $p_{1,2} = 0.75, p_{1,3} = 0.25$
Chapter 4

Incremental Self Assembly

Recall that, in incremental self assembly, polymers can grow only one monomer at a time so that, for given $N$, $p_{i,j} > 0$ if $i = 1$, and $j \leq N - 1$ and is 0 otherwise; $N$-polymers form the yield of the process. Finding explicit solutions using the approach of Chapter 3 appears difficult for general $N$ and $\{p_{i,j}\}$. Surprisingly, an interesting case that is tractable occurs when the total reaction rates are $x_1 x_j$ for all $j$. One obtains this case when $p_{1,1} = 1$ and $p_{1,j} = 1/2$ for $j > 1$. In an alternative interpretation, one can take all reaction rates to be 1 and think of the choices of a size 1 and size $j$ polymer in a self-assembly event to be proportional to the number of (unordered) pairs of such polymers. We begin this chapter with a different analysis of the triangle self assembly problem covered in the previous chapter; this is the simplest nontrivial incremental self assembly problem. We will check our computations by comparison with the earlier results. The new approach is then extended to general $N$.

4.1 Assembly of Triangles

As discussed in Chapter 3, the system of differential equations describing the rates of change of polymer concentrations in the system is
\[
\frac{dx_1(t)}{dt} = -2px_1^2(t) - x_1(t)x_2(t) \quad (4.1)
\]
\[
\frac{dx_2(t)}{dt} = x_1^2(t) - x_1(t)x_2(t) \quad (4.2)
\]

Instead of reducing these two equations to one as we did earlier, we will keep the two equations, and solve the system using the standard eigenvalue method for solving systems of ODEs.

Dividing both equations by \(x_1(t)\) and introducing a new variable \(s\) where \(ds = x_1(t)dt\), \(y_2(s) := x_2(t(s))\) and \(y_1(s) := x_1(t(s))\), we obtain the linear system

\[
\frac{dy_1}{ds} = -2y_1 - y_2 \quad (4.3)
\]
\[
\frac{dy_2}{ds} = y_1 - y_2 \quad (4.4)
\]

where \(y_1(0) = 1\) and \(y_2(0) = 0\). By standard methods, the solution to (4.3) and (4.4) is found to be

\[
y_1(s) = e^{-3s/2} \left( \cos \frac{\sqrt{3}}{2} s - \frac{\sqrt{3}}{3} \sin \frac{\sqrt{3}}{2} s \right) \quad (4.5)
\]
\[
y_2(s) = \frac{2\sqrt{3}}{3} e^{-3s/2} \sin \frac{\sqrt{3}}{2} s \quad (4.6)
\]

For the yield calculation, define a stopping time, \(s_*\), as the first time that \(y_1(s_*) = 0\) (corresponding to \(x_1(t) = 0\) – no monomers are left), i.e., \(s_* = \inf \{s : y_1(s) = 0\}\) is the time the system enters an absorbing state in the \(s\)-domain. According to (4.5)–(4.6), we find

\[
s_* = \frac{2\sqrt{3} \pi}{9}
\]

This gives us
\[ y_2(s_*) = x_2(\infty) = e^{-\sqrt{3}\pi/3} \approx 0.163033535^- \]

and a yield of

\[ \beta = 3y_3(s_*) = 1 - 2y_2(s_*) = 0.67393293^+ \]

This means that roughly a third of the monomers are wasted as dimers. To verify this result, substitute the value of \( \rho = p_{1,1}/p_{1,2} = 2 \) in (3.18) to obtain the same value of \( y_2(s_*) \), as follows.

\[
x_2(\infty) = \frac{\sqrt{\rho}}{2} \exp \left[ \left( \frac{1 + \rho}{\sqrt{\rho^2 + 4\rho - 1}} \right) \left( \arccos \left( \frac{1 - \rho}{\sqrt{2\rho}} \right) - \pi \right) \right]
= \exp \left[ \sqrt{3} \left( \arccos \left( \frac{1}{2} \right) - \pi \right) \right]
= e^{-\sqrt{3}\pi/3}
\]

### 4.2 Assembly of Systems with Arbitrary \( N \)

Using the method of the preceding section, the system of differential equations in (1.7)–(1.8) converts to a linear system of differential equations, and is given by

\[
\frac{dy_1}{ds} = -2y_1 - \sum_{k=2}^{N-2} y_k
\]

\[
\frac{dy_k}{ds} = y_{k-1} - y_k
\]

where \( ds = x_1(t)dt \) and \( k = 2, \ldots, N - 1 \). In the standard approach, let \( A \) be the coefficient matrix, and let \( I \) be the identity matrix of dimension \( N - 1 \). We first find the eigenvalues of the system from the characteristic equation. We have

\[
\det(A - \lambda I) = \det(\tilde{A} - (\lambda + 1)I) = 0
\]
CHAPTER 4. INCREMENTAL SELF ASSEMBLY

where

\[
\bar{A} = \begin{bmatrix}
-1 & -1 & -1 & \ldots & -1 \\
1 & 0 & & & \\
1 & 0 & & & \\
& & & & \\
& & & & \\
& & & & \\
\end{bmatrix}
\]  \hfill (4.8)

where the \( \lambda \) is the eigenvalue of the system. Exploiting the structure of \( \bar{A} \) and the properties of the determinant, we find that

\[
\det(A - \lambda I) = (-2 - \lambda)(-1 - \lambda)^{N-2} + (-1)(-1 - \lambda)^{N-3} + (-1)(-1 - \lambda)^{N-4} + \ldots
\]

\[
= (-1)^{N-2} \left( (-2 - \lambda)(\lambda + 1)^{N-2} - \sum_{i=0}^{N-3} (\lambda + 1)^i \right)
\]

\[
= (-1)^{N-1} \sum_{i=0}^{N-1} (\lambda + 1)^i
\]

\[
= (-1)^N \frac{(\lambda + 1)^N - 1}{(\lambda + 1) - 1}
\]

Since \( e^{i2\pi \ell} = 1 \), the characteristic roots are given by

\[
\lambda_\ell = e^{i2\pi \ell/N} - 1
\]  \hfill (4.9)

\( \ell = 1, \ldots, N - 1 \). Therefore, the \( y_k, k = 1, \ldots, N - 1 \), have the following form

\[
y_k(s) = \sum_{\ell=1}^{N-1} c_{k\ell} e^{\lambda_\ell s}
\]  \hfill (4.10)

where \( c_{k\ell} \) are coefficients to be determined. Substitution of (4.10) into (4.7) results in

\[
\sum_{\ell=1}^{N-1} \lambda_\ell c_{k\ell} e^{\lambda_\ell s} = \sum_{\ell=1}^{N-1} (c_{(k-1)\ell} - c_{k\ell}) e^{\lambda_\ell s}
\]

with \( k = 2, \ldots, N - 1 \). We obtain the recurrence \( c_{k\ell}(\lambda_\ell + 1) = c_{(k-1)\ell} \) with the solution for \( c_{k\ell} \) given by
\[ c_{k\ell} = (\lambda_{\ell} + 1)^{-k+1} c_{1\ell} \]
\[ = e^{-i2\pi\ell(k-1)/N} c_{1\ell} \]  
(4.11)

with \( \lambda_{\ell} + 1 = e^{i2\pi\ell/N} \) given by (4.9). For notational convenience, let \( c_{\ell} \equiv c_{1\ell}, \ell = 1, \ldots, N - 1 \). Combining (4.11) with (4.10), we have

\[ y_k(s) = \sum_{\ell=1}^{N-1} c_{\ell} \exp \left[ -i2\pi\ell(k-1)/N \right] e^{\lambda_{\ell}s}. \]  
(4.12)

At this point, all \( y_k \) are defined in terms of the unknown coefficients \( c_\ell \). In what follows, we use the initial conditions \( y_1(0) = 1 \) and \( y_k(0) = 0 \), where \( k = 2, \ldots, N - 1 \), to determine the coefficients \( c_\ell \). Define \( \epsilon := e^{-i2\pi/N} \); the initial conditions and (4.12) yield

\[ \sum_{\ell=1}^{N-1} c_{\ell} = 1, \]
\[ \sum_{\ell=1}^{N-1} c_{\ell} \epsilon^{(k-1)} = 0, \quad k = 2, \ldots, N - 1. \]

This system can be rewritten in the matrix form \( BC = U \) where \( U \) is a unit vector. The matrix form is given by

\[
\begin{bmatrix}
1 & 1 & 1 & 1 & \ldots & 1 \\
\epsilon & \epsilon^2 & \epsilon^3 & \epsilon^4 & \ldots & \epsilon^{N-1} \\
\epsilon^2 & \epsilon^4 & \epsilon^6 & \ldots & \epsilon^{2N-2} \\
\vdots & & & & \ddots & \\
\epsilon^{N-2} & \ldots & & & & \epsilon^{N-2N} \\
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
\vdots \\
c_{N-1} \\
\end{bmatrix}
= \begin{bmatrix}
1 \\
0 \\
0 \\
\vdots \\
0 \\
\end{bmatrix}
\]

By Cramer's rule, the solution is given by

\[ c_\ell = \frac{\det B_\ell}{\det B} \]  
(4.13)
where $B_\ell$ is obtained from $B$ by replacing the $\ell$-th column with $U$. It can be shown that, because of the form of $B$, one has

$$\det B = \prod_{1 \leq j < k \leq N-1} (\epsilon^k - \epsilon^j)$$

(4.14)

and

$$\det B_\ell = (-1)^{\ell+1} \begin{vmatrix} \epsilon & \epsilon^2 & \ldots & \epsilon^{\ell-1} & \epsilon^{\ell+1} & \ldots & \epsilon^{N-1} \\ \epsilon^2 & \epsilon^4 & \ldots \\ \vdots \\ \epsilon^{N-2} \end{vmatrix} = (-1)^{\ell+1} \epsilon^{\frac{N(N-1)}{2} - \ell} \prod_{1 \leq j < k \leq N-1} (\epsilon^k - \epsilon^j)$$

Recall that $\epsilon = e^{-i2\pi/N}$, which in conjunction with the preceding formula, implies

$$\det B_\ell = (-1)^{N+\ell} \epsilon^{-\ell} \prod_{1 \leq j < k \leq N-1} (\epsilon^k - \epsilon^j)$$

(4.15)

Substituting (4.14) and (4.15) into (4.13) and recalling that $e^{i2\pi\ell} = 1$, one obtains

$$c_\ell = (-1)^{N-1} \epsilon^{-\ell} \left[ \prod_{j \neq \ell} (\epsilon^j - \epsilon^\ell) \right]^{-1} = (-1)^{N-1} (1 - \epsilon^\ell) \left[ \prod_{1 \leq j \leq N-1} (\epsilon^j - 1) \right]^{-1}$$

where, to get the last expression, we used the properties of $\epsilon = e^{-i2\pi/N}$. Hence, in view of (4.12)
\[ y_k(s) = \left[ \prod_{1 \leq j \leq N-1} (1 - \epsilon^j) \right]^{-1} \sum_{\ell=1}^{N-1} (\epsilon^{-\ell} - 1) \epsilon^{\ell k} e^{\lambda_\ell s} \]

Since \( y_1(0) = 1 \), the continued product must evaluate to \( N \), so

\[ y_k(s) = \frac{1}{N} \sum_{\ell=1}^{N-1} (\epsilon^{-\ell} - 1) \epsilon^{\ell k} e^{\lambda_\ell s} \]  \hspace{1cm} (4.16)

Now we have an explicit solution for \( y_k \). For the yield calculation, recall from the previous section that the stopping time, \( s_\ast \), is the first time that \( y_1(s_\ast) = 0 \) (corresponding to \( x_1(t) = 0 \), no monomers are left), i.e., \( s_\ast = \inf \{ s : y_1(s) = 0 \} \) is the time the system enters an absorbing state in the \( s \)-domain. In view of (1.6), at time \( s_\ast \), the number of monomers used to generate \( y_k \), where \( k = 1, \ldots, N-1 \), in the \( s \)-domain is \( \sum_{k=1}^{N-1} k y_k(s_\ast) \). Therefore, the yield, \( \beta = \beta(N) \), is given by

\[
\beta = N y_N(s_\ast) = 1 - \sum_{k=1}^{N-1} k y_k(s_\ast) \\
= 1 - \frac{N}{\lambda_0} \sum_{\ell=1}^{N-1} \frac{\epsilon^{-\ell} - 1}{\epsilon^{\ell} - 1} e^{\lambda_\ell s_\ast} \\
= 1 + \sum_{\ell=1}^{N-1} e^{-\ell} e^{\lambda_\ell s_\ast} \\
= \sum_{\ell=0}^{N-1} e^{-\ell} e^{\lambda_\ell s_\ast}
\]

where \( \lambda_0 = 0 \), and \( \lambda_\ell = e^{2\pi \ell/N} - 1 \) are the characteristic roots given by (4.9).

Using Taylor’s expansion, we find that

\[
\beta = e^{-s_\ast} \sum_{\ell=0}^{N-1} e^{-\ell} \sum_{k=0}^{\infty} \frac{e^{-k\ell} s_\ast^k}{k!} \\
= e^{-s_\ast} \sum_{k=0}^{\infty} \frac{s_\ast^k}{k!} \sum_{\ell=0}^{N-1} e^{-\ell(k+1)}
\]

Next, we observe that
\[ \sum_{\ell=0}^{N-1} e^{-\ell k} = \sum_{\ell=0}^{N-1} e^{\ell k} = \begin{cases} N & k = mN, m \in \mathbb{Z} \\ 0 & \text{otherwise} \end{cases} \quad (4.17) \]

and hence

\[ \beta = e^{-s_*} \sum_{k=1}^{\infty} \frac{N s_*^{kN-1}}{(kN - 1)!} \quad (4.18) \]

In (4.16), set \( k = 1, s = s_* \) and use \( y_1(s_*) = 0 \) to obtain

\[ 0 = \frac{1}{N} e^{-s_*} \sum_{\ell=0}^{N-1} (1 - \epsilon^\ell) e^{-t s_*} \]
\[ = \frac{1}{N} e^{-s_*} \sum_{k=0}^{\infty} \frac{s_*^k}{k!} \sum_{\ell=1}^{N-1} [e^{-\ell k} - e^{-\ell(k-1)}] \quad (4.19) \]

In view of (4.17), \( s_* \) satisfies

\[ 1 - s_* + \sum_{k=1}^{\infty} \frac{s_*^{kN}}{(kN)!} \left( 1 - \frac{s_*}{kN + 1} \right) = 0 \quad (4.20) \]

We can compute \( s_* \) numerically from this equation and substitute it into (4.18) to obtain the yield \( \beta \).

Consider next asymptotics in \( N \). Since \( s_* \) is an integral of \( x_1(t) \), then \( s_* = s_*(N) \) is a nondecreasing function of \( N \). This coupled with (4.20) leads to

\[ s_*(N) = 1 + O\left(\frac{1}{N!}\right) \]

which gives

\[ \lim_{N \to \infty} \beta(N) = \frac{N^2}{N!} (1 + o(1)) \]

As a partial check, consider the system of triangle self assembly. We find that \( s_* = 2\sqrt{3}\pi/9 \) found in the previous section satisfies (4.20) for \( N = 3 \). Substituting this \( s_* \) and \( N = 3 \) into (4.18), we obtain \( \beta = 0.67393 \ldots \) as before.
Chapter 5

Reversible Self Assembly

Reversible self assembly also allows the decomposition of polymers into two constituent polymers, e.g., a $k$-polymer may decompose into $i$- and $j$-polymers, $i + j = k$, and be governed by given rates. If $N$-polymers precipitate as yield, then it is generally assumed that $N$-polymers cannot be broken down and that they have monotone nondecreasing concentrations as in the earlier models. So while the model no longer exhibits pure growth, it remains a growth model. As before, the reversible system can be expressed as a system of differential equations comprising a special case of Feinberg’s equations [27] for describing chemical kinetics; added terms take the form $q_{i,k-i}x_k$ with $q_{i,k-i}$ a decomposition rate analogous to reaction rates. For example, reversible incremental self assembly is described by the following obvious extension of (1.7)

\[
\frac{dx_i(t)}{dt} = 2x_1(t)x_{i-1}(t)p_{i-1} - 2x_1(t)x_i(t)p_i + x_{i+1}(t)q_i - x_i(t)q_{i-1}, \ i > 2
\]

\[
= x^2_1(t)p_1 - 2x_1(t)x_i(t)p_i + x_{i+1}(t)q_i - x_i(t)q_{i-1}, \ i = 2
\]

\[
= -2 \sum_{l \geq 1} p_1 x_1(t)x_l(t) + 2x_2(t)q_1 - \sum_{j \geq 2} x_{j+1}(t)q_j, \ i = 1
\]  

(5.1)

where $q_j$ is the probability that a $(j + 1)$-polymer decomposes into a monomer and a $j$-polymer. In general, the problems of solving the reaction rate equations become
even more difficult, so explicit solutions can be expected only for rather special cases. Simulation tools remain as the principal recourse in studies of reversible systems. To extend the derivation in Chapter 2, we define, in addition to \( R(t) \), a total rate \( D(t) \) of polymer decompositions which is proportional to the concentration of reactant polymers. In particular, in analogy with reactions, an elementary model stipulates that \( D(t) = \mu \eta(t) \), where, as before, the constant \( \mu \) accounts for physical parameters affecting decompositions, and where the intervals between successive decompositions are exponentially distributed with the rate parameter \( D(t) \). Hence, events occur at a total rate of \( R(t) + D(t) \), and the times between successive events are modeled as exponentials with mean \( 1/(R(t) + D(t)) \). The general incremental polymerization equations have the form,

\[
\frac{d\eta_i(t)}{dt} = \begin{cases} 
2\eta_i(t)(\eta_{i-1}(t)p_{i-1} - \eta_{i+1}(t)p_i) + \eta_{i+1}(t)q_i - \eta_i(t)q_{i-1} & i > 2, \\
\eta_1^2(t)p_1 - 2\eta_1(t)\eta_2(t)p_2 + \eta_{i+1}(t)q_i - \eta_i(t)q_{i-1} & i = 2
\end{cases}
\] (5.2)

where the constant \( \mu \) has been absorbed by the \( q_i \) parameters. The limit under the LLN scaling again gives the appropriate reaction rate equations in (5.1), as expected.

As a companion to the examples verifying accuracy given in Chapter 2, the results of a simulation of reversible self assembly are compared in Figure 5.1 with the numerical solution of the reaction rate equations. The figure shows the evolution of \( \eta_2(t) \) for \( N = 3 \) and \( N = 4 \).

It is important to note that yields can be much higher in systems with polymer decomposition. We illustrate this notion within an interesting problem of minimizing self assembly times assuming that decomposition rates can be adjusted as functions of time (e.g., by increasing or decreasing ambient temperatures). For the sake of analytical tractability, we consider a triangle-type self assembly model with the single parameter \( q_{1,1} \equiv q_{1,1}(t) \) by which dimers decompose into two monomers). Specifically,
CHAPTER 5. REVERSIBLE SELF ASSEMBLY

Figure 5.1: Plots for the simulation and numerical results for $\eta_2(t)$ with $N = 3, 4$ for reversible self-assembly. The bonding and decomposition probabilities are given on the top of each figure. We start this experiment with $\eta(0) = 5,000$ number of monomers.

we extend as follows the incremental system of Chapter II.4 for $N = 3$, $p_{1,1} = 1$, and $p_{1,2} = 1/2$.

Using arguments similar to those in Chapter II.4, the system of differential equations for this system is given by (again, the dependence on $t$ is suppressed for compactness)

$$
\frac{dx_1}{dt} = -2x_1^2 - x_1 x_2 + 2q_{1,1}x_2 \\
\frac{dx_2}{dt} = x_1^2 - x_1 x_2 - q_{1,1}x_2 \\
\frac{dx_3}{dt} = x_1 x_2
$$

The final terms in the first two equations reflect, respectively, the addition of two monomers, and the loss of a dimer when a dimer decomposes.

The problem is to define $q_{1,1}(t)$ so that the system achieves 100% yield in minimal time. Analytically, this translates to specifying $q_{1,1}$ so that the system stays on its
eigenvectors after it reaches them\(^1\), say at time \(\tau\). At this time, the concentration of monomers in the system must be twice that of the dimers, because the system must exhaust all resources when it reaches an absorbing state along the eigenvectors. Now consider the evolution of the system beyond time \(\tau\). For simplicity, translate time by \(\tau\) and make the initial state the one existing when the system reaches the eigenvectors. Equation (5.3) can be solved with

\[
\frac{dx_1}{dt} = 2\frac{dx_2}{dt}
\]

with \(x_1(0) = 2K\) and \(x_2(0) = K\), where \(K\) is a constant to be determined. One arrives at

\[
\frac{dx_1}{dt} = 2\frac{dx_2}{dt}
\]

\[-2x_1^2 - x_1x_2 + 2q_{1,1}x_2 = 2x_2^2 - 2x_1x_2 - 2q_{1,1}x_2
\]

\[-4x_1^2 + x_1x_2 = -4q_{1,1}x_2
\]

\[-x_1^2 + \frac{x_1x_2}{4} = -q_{1,1}x_2
\]

Applying the initial conditions, \(x_1(0) = 2K\) and \(x_2(0) = K\), gives

\[-4x_2^2 + \frac{2x_2^2}{4} = -q_{1,1}x_2
\]

\[q_{1,1} = \frac{7x_2}{2}
\]

from which,

\[x_1 = 2x_2 = \frac{4q_{1,1}}{7}
\]

One can obtain \(x_1(t)\) by solving

\[
\frac{dx_1}{dt} = -2x_1^2 - \frac{x_2^2}{2}
\]

\(^1\)See the discussion at the end of Chapter II.3 in dealing with phase transitions.
which yields

\[ x_1(t) = 2 \left( \frac{3t}{2} + \frac{1}{K} \right)^{-1} = 2x_2(t) \]

Using the property of \( \sum i x_i = 1 \) and the conditions \( x_3(0) = 0, x_3(\infty) = K \), we find that

\[ x_3(t) = \frac{4K}{3} - \frac{4}{3} \left( \frac{3t}{2} + \frac{1}{K} \right)^{-1} \]

Hence,

\[ q_{1,1}(t) = \frac{7/2 \cdot 1_{t > \tau}}{3(t - \tau)/2 + 1/K} \]

where \( \tau := \inf\{t > 0 : x_1 = 2x_2\} \) and \( x_2(\tau) = K \) must be determined numerically. In words, the system is allowed to evolve (absent control) from the initial state \( x_1(0) = 1 \) until the concentration of monomers first becomes exactly twice that of the dimers (at time \( \tau \)). At this point, the control is exerted as above so as to preserve the relative concentrations of monomers and dimers.
Part III

Tile Self Assembly
Chapter 1

Introduction

In contrast to the abstract model of molecular self assembly in Part II, the current Part III introduces a concrete model based on molecules created from DNA; this widely studied self-assembly model is called the tile model of self-assembly, and, again in contrast to Part II, it leads to an explicit model of computation. A tile is an abstraction of block elements used in the Block Cellular Automata (BCA) discussed in Part I. Tiles can self-assemble according to an appropriate set of rules in such a way that tiling patterns carry out computations.

Whereas earlier analysis of the performance of tile self-assembly focused on complexity and optimization issues, the work here focuses on the time to complete the self-assembly of rectangular arrays of tiles, as introduced in Chapter I.1. Throughout Part III, we assume that self assembly follows the rule-set structure given in Chapter I.1, where the Tile Assembly Model (TAM) is defined.

The performance of linear tile self-assembly was studied by Adleman et al [1]. In their model, there are \( n \) tile types arranged on a doubly infinite line. These tiles bond to their neighbors with given probabilities, \( \sigma_{ij} \) 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
CHAPTER 1. INTRODUCTION

broken with probability \( \tau_{ij} \), 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.

They studied the problem of \( n \)-linear polymerization, where the system consists of equal quantities of tiles \( T_1, \ldots, T_n \), 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 given probabilities \( \sigma_{ij}, 1 \leq i, j \leq n \). 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 probability \( \sigma_{ij} \), and supertiles are allowed to be broken with probability \( \tau_{ij} \). 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 two dimensional self-assembly models, such as the tile model introduced by Winfree et al [62] as an adaptation of Wang’s [55] original tile model of computation. Part III studies computing times in the hydrodynamic limit of Winfree’s model with pre-assembled border tiles and growing border tiles in Chapter 2. Since the scale of molecular self assembly is so large, passing from the discrete setting to hydrodynamic limits has been shown repeatedly to introduce negligible error.

The analysis of computing times leads to estimates of shapes and other performance metrics of tile self-assembly. However, analysis carries one only so far; further
insights must rely on experimental studies based on the simulation of the dynamics of discrete molecular systems. Experiments are presented and discussed in Chapter 2. In addition to containing more general characterizations of behavior, this chapter verifies, where explicit analytical results are possible for hydrodynamic limits, the exceptionally close agreement between these results and the experimental results of discrete, molecular-scale systems.

In subsequent chapters, we discuss the difficulties of deploying tile self-assembly applications in the laboratory. Winfree et al introduced a pioneering tile self-assembly application using DNA molecules abundant in nature [62]. However, tile self-assembly using DNA is error prone. In this respect, the current stage of development in tile self-assembly parallels the early development of silicon-based computing. As a result, error-correction or fault-tolerant techniques are needed. In Chapter 4, we discuss various error-correction methods including a pulsing method which varies temperature to control, and exploit, the error correction realized by the reversibility of self-assembly.

In the remainder of this chapter, we introduce two well known interacting particle processes in applied probability called the Totally Asymmetric Simple Exclusion Process (TASEP) and the Hammersley process. Some important results drawn from the analysis of these processes, such as the passage time of TASEP and the expected length of longest up-right paths in the graphical representations of the Hammersley process, will be applied in the analysis of tile self-assembly to be discussed later in Part III. Covering this material here avoids later disruptions in the flow of the analysis of DNA-based computing times. Full details of the proofs of the results needed here fall well beyond the scope of this thesis. However, our interest is confined primarily to hydrodynamic limits for which useful heuristic arguments exist. These avoid lengthy technicalities, and are meant to give basic insights on which proofs can be based.
1.1 The Totally Asymmetric Simple Exclusion Process

The title process has become a paradigm for nonreversible stochastic particle systems. The states of the Totally Asymmetric Simple Exclusion Process (TASEP) are binary strings, with a 1 in position $j$ meaning that a particle occupies the $j$-th “slot” $[j-1,j]$, and a 0 meaning that the $j$-th slot is unoccupied (or that a “hole” occupies the $j$-th slot). The evolution of the process is governed by the exchange rule: Independently, each particle waits for an exponentially distributed time with mean 1, and then changes places with the hole in the slot to its immediate right, provided that hole exists. If the slot on the immediate right is occupied, the particle does not move; in either case, a new exponential waiting time commences and the process repeats.

![Figure 1.1: The initial jump of the TASEP and the two possibilities for the second jump.](image)

In the TASEP of interest here, particles are initially placed in the *megajam* configuration, where all slots to the left of the origin are occupied by particles, and all slots to the right are unoccupied. Index the particles leftward so that the first one is in the right-most occupied slot, but index the holes in the rightward direction, i.e., the first one is the left-most unoccupied slot. Figure 1.1 shows, on the left, a single (the only allowable) particle-hole exchange of the TASEP initially in the megajam configuration; and on the right, the arrows represent the two possible subsequent exchanges.

Our primary objective in this section is a expected-value estimate of the time taken for the $M$-th hole to be exchanged with the $N$-th particle, which we denote by $C_{M,N}$. This passage time satisfies the recurrence
\[ C_{M,N} = \max\{C_{M-1,N}, C_{M,N-1}\} + T_{M,N} \]

where \( T_{M,N} \) is the time it takes for the \( M \)-th hole to exchange slots with the \( N \)-th particle immediately to its left; the \( T_{i,j} \) are taken to be i. i. d exponentially distributed random variables with unit mean. To see how one arrives at this recurrence, note that before the \( M \)-th hole can exchange with the \( N \)-th particle, the \((M - 1)\)-st hole must have exchanged with the \( N \)-th particle and the \((N - 1)\)-st particle must have exchanged with the \( M \)-th hole. When the later of these two events occurs, the \( M \)-th hole is to the immediate right of the \( N \)-th particle, so after a wait of \( T_{M,N} \) the two will exchange places. The solution to this recurrence can be expressed in terms of paths \( \omega \) from \((1,1)\) to \((M,N)\) in which each successive lattice point differs from the previous point by 1 in one coordinate or the other, but not both, each step in the path is a single step to the right or a single step upward; in these terms it is easily verified that

\[ C_{M,N} = \max_{\omega: (1,1) \rightarrow (M,N)} \sum_{(i,j) \in \omega} T_{i,j} \]

where the max is taken over the set of nondecreasing paths from \((1,1)\) to \((M,N)\).

(For a proof of this assertion, see Section 2.1.)

In what follows, an estimate of \( C_{M,N} \) is obtained from an analysis of the hydrodynamic limit [48, 19, 9] of the TASEP. In the development of this limit, it is convenient to redefine the TASEP in the obvious way on consecutive slots of length \( 1/n \), where \( n \) is a scaling parameter. At time \( t \), consider a small interval \([x, x+\Delta x]\) that, for simplicity, begins and ends at slot boundaries as shown in Figure 1.3; but take \( n \) so that there are a large number of slots in \([x, x+\Delta x]\), that is, take \( n \) so that \( n\Delta x \gg 1 \). Classical analysis shows that the Bernoulli configurations form stationary distributions for TASEP. Hence, heuristically after a long time \( t \), the TASEP particles in \([x, x+\Delta x]\) at time \( t \) comprise a partial trajectory of a process that is approximately Bernoulli with
a (probability) parameter \( p(x, t) \) which depends on \( x \) but is approximately constant over \([x, x + \Delta x]\) for small \( \Delta x \). If we consider the TASEP in statistical equilibrium, then as the continuous limit of small \( \Delta x \), small \( 1/n \), but large \( n \Delta x \) is approached, the number of particles \( p(x, t) \Delta x \) scaled by \( n \) approaches the particle mass \( \rho(x, t) dx \) where \( \rho(x, t) \) is a density measured in units of mass per unit distance. Further, in the Bernoulli process, the particle can move one step with probability \( 1 - \rho(x, t) \) and stays at the same place with probability \( \rho(x, t) \). Hence, at point \( x \) at time \( t \), the average particle velocity approaches a velocity of particle mass, \( v(x, t) := 1 - \rho(x, t) \), which can be seen to be the limiting density corresponding to the hole probability of the Bernoulli process.

With this set-up, we now develop Burgers equation for the hydrodynamic limit of the TASEP in equilibrium. Define the flux of the limiting process past point \( x \in \mathbb{R} \) at time \( t \) as \( f(x, t) := \rho(x, t) v(x, t) \), and note that it measures flow in units of mass per unit time. Express the change in flux \( f(x, t) - f(x + \Delta x, t) \) over an interval \([x, x + \Delta x]\) of the real line in terms of the density \( \frac{\partial \rho}{\partial t} \) measuring flow in units of mass per unit time per unit distance. One obtains, within a term on the order of \( o(\Delta x) \),

\[
f(x, t) - f(x + \Delta x, t) = \int_x^{x+\Delta x} \frac{\partial \rho(x, t)}{\partial t} dx \approx \frac{\partial \rho(x, t)}{\partial t} \Delta x
\]

and hence, suppressing the dependence on \( x, t \),

\[
\frac{\partial f}{\partial x} = -\frac{\partial \rho}{\partial t}
\]

whereupon substitution of \( f = \rho v = \rho(1 - \rho) \) gives Burgers equation

\[
\frac{\partial}{\partial x} [\rho(1 - \rho)] + \frac{\partial \rho}{\partial t} = 0 \tag{1.1}
\]

The density of particle mass corresponding to the initial megajam configuration of the TASEP is a step function, where the step is from 1 to 0 at the origin. With this initial condition, a solution to Burger’s equation (1.1) gives the linear densities
Figure 1.2: Fluid limit representation of TASEP and its underlying process.

Figure 1.3: The TASEP initially at megajam configuration can be analyzed using Burgers equation. The density of particle $x$ at time $t$ is represented by $u(x,t)$, which changes as $t$ increases.

illustrated in Figure 1.3. The integral of the density over region $A$ gives a particle mass which, in the discrete TASEP, approximates a number, say $N$, of particles.
Similarly, the integral of the hole density, \(1 - \rho(x, t)\), over region \(B\) approximates a number, say \(M\), of holes in the discrete TASEP. The \(M\) holes are to the left of the \(N\) particles, so consistent with our indexing convention (particles right-to-left and holes left-to-right), \(z\) in the figure can be interpreted as corresponding to the location where the \(N\)-th particle and the \(M\)-th hole are in adjacent slots and hence where their exchange takes place; and so the time \(t\) is an approximation of \(C_{M,N}\). Thus,

\[
M \sim \frac{(t - z)^2}{4t} \\
N \sim \frac{(t + z)^2}{4t}
\]

Eliminating \(z\) and solving for \(t \sim C_{M,N}\), one obtains the desired result

\[
C_{M,N} \sim (\sqrt{M} + \sqrt{N})^2
\] (1.2)

Precisely, (1.2) holds asymptotically as \(M, N \to \infty\) with \(M/N\) held constant.

For arbitrary non-negative reals \(x\) and \(y\), the continuous hydrodynamic limit of \(C_{M,N}\) is defined as

\[
C_{x,y} = \lim_{n \to \infty} \frac{1}{n} C_{[nx],[ny]} 
\] (1.3)

and is given by a similar equation [39]

\[
C_{x,y} = (\sqrt{x} + \sqrt{y})^2
\] (1.4)

Prähofer and Spohn [45] showed that the fluctuations of the passage time of the TASEP from \((0, 0)\) to \((N, N)\) have a standard deviation on the order of \(N^{1/3}\). When \(N\) is large, therefore, the average computing times obtained from simulation experiments (see Chapter 2) are in very close agreement with the asymptotic estimates in (1.2) and (1.4).
1.2 Hammersley’s process

Implicit in the Hammersley process is a classic algorithm [29] for the problem of finding the length of a longest increasing subsequence in a given sequence of numbers. To illustrate the problem, note that the sequence \( \pi_n = \{7, 6, 2, 5, 4, 9, 10, 8\} \) has longest increasing subsequences 2, 5, 9, 10 and 2, 4, 9, 10. In the average-case version of the problem of interest here, one is given a (uniform) random ordering of a given set of numbers and is asked for the expected length of a longest increasing subsequence. Our objective here is to give a heuristic, but convincing argument that the answer is \( 2\sqrt{n} \), where \( n \) is the length of the given sequence. This argument is due to Aldous and Diaconis [7].

The interacting-particle, Poissonized version of the Hammersley process has states consisting of particle configurations on \( \mathbb{R}^+ \); the particle positions in the interval \([0, x_*]\) evolve as follows:

At the epochs of a rate-\( x_* \) Poisson arrival process, points are chosen uniformly at random on \([0, x_*]\). If \( x_i \) is the \( i \)-th point chosen, and if there exists a particle to the right of \( x_i \), then the closest such particle is moved to position \( x_i \); otherwise, if no such particle exists, a new particle is created and placed at \( x_i \).

The particle trajectories in a space-time graphical representation of the Hammersley process are illustrated in Figure 1.4; the abscissas of the corners of the L shaped segments are the points \( x_i \) chosen in \([0, x_*]\). The horizontal and vertical line segments represent the trajectory of the particles.

At time 0, the Hammersley process begins with no particles in \( \mathbb{R}^+ \). The first \( n \) points chosen in \([0, x_*]\) constitute a random permutation \( \pi_n \) of \( n \) numbers, where \( \pi_n \) is the sequence of times that particles last moved, and, as is easily verified, the number
of particles just after the n-th choice is the length of a longest increasing subsequence in \( \pi_n \) \[29\]. Now take \( x_* = n \) and run the process for one time unit starting in the initial state. If \( N \) denotes the Poisson-distributed number of arrivals, then \( \mathbb{E} N = n \) and, as argued below, the expected length of the longest increasing subsequences of \( \pi_N \) is asymptotically \( 2\sqrt{n} \) for large \( n \). A dePoissonization is needed to obtain this result with the number of arrivals fixed at \( n \). These details are omitted not only because they apply routine techniques, but because our application uses the result for the Poissonized version of the problem.

For present purposes, arguments about the Hammersley process are most conveniently expressed in terms of the Hammersley counting process, \( N(x,t) \), the number of particles in \([0,x]\) at time \( t \). Indeed, the \( \sqrt{n} \) asymptotic growth rate, \( r(t) \), is easily proved from classical results of subadditive processes. For, if \( r(t) := \mathbb{E} N(t,t) \), then by considering paths of points from \((0,0)\) to \((t+s,t+s)\) via \((t,t)\), we find without
difficulty that

\[ r(t + s) \geq r(t) + r(s), \quad s, t \geq 0 \]

That is, \( r(t) \) is superadditive, and hence an application of the subadditive ergodic theorem shows that there exists a constant \( c \) such that \( \mathbb{E}N(t, t) \sim c\sqrt{t} \) as \( n \to \infty \). The proof that \( c = 2 \), amounting originally to hard combinatorial arguments \([42]\), requires much more effort.

To continue the heuristic argument within the particle-process formulation, consider the configuration of particles in \([0, x]\) at time \( t \). The numbers \( x_i \), chosen in \([0, x]\) form a Poisson pattern, and it is tempting to try to prove that the particles created and translated by the Hammersley process preserve the Poisson properties, restricted perhaps to the asymptotic regime of large \( x \). With this in mind, suppose the particle positions are approximately the points of a Poisson process for large \( x \), and let \( \lambda(x, t) \) denote the instantaneous rate at point \( x \). By definition, one observes that, if events with probabilities \( o(\Delta t) \) are ignored, then \( N(x, t + \Delta t) - N(x, t) = 1 \) if and only if an arrival in \([t, t + \Delta t]\) chooses a point between \( x \) and the nearest particle to the left of \( x \); otherwise, the count does not change during \([t, t + \Delta t]\). Then, by the uniform distribution governing the points \( x_i \),

\[
\frac{\partial}{\partial t} \mathbb{E}N(x, t) = \mathbb{E}L(x, t),
\]

where \( L(x, t) \) is the distance between \( x \) and the nearest particle to the left of \( x \).

But, under the Poisson approximation for particle configurations,

\[
\mathbb{E}L(x, t) \approx \frac{1}{\lambda(x, t)} \approx \frac{1}{\frac{\partial}{\partial x} \mathbb{E}N(x, t)}
\]

and so \( r = \mathbb{E}N(x, t) \) satisfies approximately the partial differential equation (PDE)

\[
\frac{\partial r}{\partial t} + \frac{\partial r}{\partial x} = 1, \quad r(0, t) = r(x, 0) = 0
\]
CHAPTER 1. INTRODUCTION

The solution to this PDE is \( r = 2\sqrt{xt} \), so with \( t = 1, x = n \), the expected length \( r_n = r(n, 1) \) of a longest increasing subsequence of an \( N \) element sequence, where \( N \) is Poisson distributed with mean \( n \), approximately satisfies

\[
r(n, 1) = 2\sqrt{n}
\]

Precisely, Aldous and Diaconis prove that, as the hydrodynamic limit is approached, the Poisson approximation becomes progressively closer to being exact, and in particular that

\[
r_n \sim 2\sqrt{n} \quad (1.5)
\]

as \( n \to \infty \).
Chapter 2

Computing Times

2.1 Pre-assembled Borders

The abstract model of tile self-assembly introduced in Chapter I.1 is studied in depth in this chapter. Reviewing briefly, each self-assembly element is modeled as a tile, of which there are three classes: rule tiles, border tiles, and seed tiles. The tile functions are illustrated in Figure 2.1. The initial, lower-left corner tile is the unique seed tile of the self assembly process. The attachments of border tiles along the horizontal and vertical boundaries of the self-assembly region are shown directly to the right of the seed tile and above it. As can be seen in the figure, rule tiles attach to available sites, which are just those unoccupied corner sites adjacent to occupied sites both to the left and below. The sequence of tile edges (top and right-hand edges) that separate the tiled region from the untiled region at time $t$ is called the profile of the process at time $t$.

In this chapter, we first investigate tile self-assembly when the borders are pre-assembled. The initial seed structure of the self-assembly is therefore an $L$-shaped input of border tiles. Only rule tiles are involved in the self-assembly process. In what follows, we assume that the attachment times of tiles of each rule type are in-
dependent and identically distributed. Rule tiles are engineered in such a way that rule tiles of each type have the same bonding strength and concentration. With these assumptions, all rule tiles can be regarded simply as indistinguishable squares. Figure 2.1 shows a cartoon of the simplified self-assembly system.

As is standard in models of particle processes (see e.g., [38, 39] and for DNA self assembly processes in particular, see e.g., Winfree [56]), the individual attachment times of rule tiles are taken to be i.i.d. exponential random variables. For convenience, the rate parameter is set at 1. Under this Markovian assumption the profile at time $t$ is a complete state at time $t$. The borders are assumed to be pre-assembled, and the problem is to estimate the expectation of the time $C_{M,N}$ necessary for a rule
Figure 2.2: A simplified version of two-dimensional tile self-assembly. The \( L \)-shape input is prefabricated, and the assembly of the rule tiles amounts to the assembly of simple square blocks. In particular, the matching edges of the tiles can be disregarded, since we focus only on the growth process.

Tile to attach to the site at \((M,N)\). Because our interest lies in molecular processes, we want estimates for large \(M\) and \(N\). Note that in order for a tile attachment to be possible at position \((M,N)\), the left (vertical) border must contain at least \(N\) tiles and the lower (horizontal) border must have at least \(M\) tiles; the lengths of the borders are otherwise immaterial. Argue as follows to obtain a formula for \(C_{M,N}\).

In accordance with the tile attachment rules, a tile can attach to \((i,j)\); \(i,j \geq 1\) when there are already tiles to the left and below \((i,j)\), in particular, tiles at \((i-1,j)\) and \((i,j-1)\). Thus, site \((i,j)\) becomes available at the larger of \(C_{i-1,j}\), the time to place the tile to the left of \((i,j)\), and \(C_{i,j-1}\), the time to place the tile below \((i,j)\). From the larger of these two times, the site remains available for a time \(T_{i,j}\) waiting to be occupied by an arriving tile. Then
\[ C_{i,j} = \max(C_{i-1,j}, C_{i,j-1}) + T_{i,j}, \quad i, j \geq 1 \]  

(2.1)

with each \( T_{i,j} \) being an independent sample from the rate-1 exponential distribution, and with the initial conditions \( C_{i,0} = C_{0,j} = 0 \) for all \( i, j \). The solution to the above recurrence is most conveniently expressed in terms of up-right paths, which we call simply paths, defined as sequences of tile placement positions starting from the initial available site at \((1,1)\). Accordingly, the path \( P : (1,1) \rightarrow (I,J) \) denotes a sequence of tile positions \( \{(i_1,j_1), (i_2,j_2), \ldots, (i_{\ell+1},j_{\ell+1})\} \) such that, for all \( \ell, 1 \leq \ell < I + J \), either \( i_{\ell+1} = i_\ell + 1 \) and \( j_{\ell+1} = j_\ell \) or \( i_{\ell+1} = i_\ell \) and \( j_{\ell+1} = j_\ell + 1 \). The lengths of these paths are all \( I + J - 1 \). It is readily proved by induction on the length of paths from \((1,1)\) to \((i,j)\) that

**Proposition 1:** The solution to (2.1) is given by

\[ C_{i,j} = \max_{P : (1,1) \rightarrow (i,j)} \sum_{(r,s) \in P} T_{r,s} \]  

(2.2)

□

**Proof:** The basis refers to \( C_{1,1} \) and a path with length 1 consisting only of \( T_{1,1} \). Thus, the basis follows immediately from (2.1) and the boundary conditions \( C_{i,0} = C_{0,j} \). Considering \( C_{i,j} \) for tiles other than the seed tile, we have by the inductive hypothesis, since the lengths of the paths from \((1,1)\) to either \((i,j-1)\) or \((i-1,j)\) are all one less than the lengths of those from \((1,1)\) to \((i,j)\), we can substitute (2.2) for \( C_{i-1,j} \) and \( C_{i,j-1} \) in (2.1) to obtain
\[ C_{i,j} = \max \left\{ \max_{\pi : (1,1) \rightarrow (i-1,j)} \sum_{(r,s) \in \mathcal{P}} T_{r,s}, \max_{\pi : (1,1) \rightarrow (i,j-1)} \sum_{(r,s) \in \mathcal{P}} T_{r,s} \right\} + T_{i,j} \]

\[ = \max \left\{ \max_{\pi : (1,1) \rightarrow (i-1,j)} \sum_{(r,s) \in \mathcal{P}} T_{r,s} + T_{i,j}, \max_{\pi : (1,1) \rightarrow (i,j-1)} \sum_{(r,s) \in \mathcal{P}} T_{r,s} + T_{i,j} \right\} \]

\[ = \max \left\{ \max_{\pi : (1,1) \rightarrow (i-1,j) \rightarrow (i,j)} \sum_{(r,s) \in \mathcal{P}} T_{r,s}, \max_{\pi : (1,1) \rightarrow (i,j-1) \rightarrow (i,j)} \sum_{(r,s) \in \mathcal{P}} T_{r,s} \right\} \]

with the first component within braces referring to paths from (1, 1) to (i, j) going through (i - 1, j) and the second component to paths from (1, 1) to (i, j) going through (i, j - 1). All paths from (1, 1) to (i, j) must go through either (i - 1, j) or (i, j - 1), and so the above expression reduces to (2.2). \[ \Box \]

2.1.1 The TASEP Model of Tile Self-Assembly

By the rules of tile placement, finite borders, say from (0, 0) to (0, N) and to (M, 0), which we regard as the 0-th column and 0-th row, respectively, constrain the self-assembly process to the rectangle defined by the above three vertices and the vertex at (M, N). In unconstrained self assembly, the boundaries are taken to be unbounded, i.e., \( M, N \rightarrow \infty \) in the above notation. For our purposes, nothing is lost by focusing on the unconstrained version; as already noted, \( C_{M,N} \) remains well defined and has the same value for all cases where the left boundary is at least \( N \) tiles tall, and the lower boundary is at least \( M \) tiles wide.

But for unconstrained self assembly, we can make a key observation: The process is isomorphic to the TASEP analyzed in Chapter III.1. Precisely, the attachment of a new tile to a self-assembly profile maps into the jump of a particle in the TASEP. The particle configuration of TASEP corresponding to a profile of (unconstrained) self assembly is obtained by labeling every horizontal tile edge of the profile with a
circle denoting a hole and every vertical tile edge of the profile with a bullet denoting a particle. Figure 2.3 illustrates the mapping.

Figure 2.3: Relationship between tile self-assembly and the TASEP.

Figure 2.3 shows the mapping of attachments in the self-assembly process onto the particle jumps in the TASEP. In part (b) of the figure, a tile is ready to attach to an available site. Part (c) shows the profile and particle configuration after the attachment. Note that the particle configuration of part (c) differs from that of part (b) by the exchange of a particle with an adjacent hole. More generally, there is a one-to-one correspondence between available (i.e., corner) sites and instances in the corresponding TASEP particle configuration where a particle is followed by a hole.
CHAPTER 2. COMPUTING TIMES

The self-assembly process starts with no rule tiles attached anywhere so the initial profile consists simply of the vertical and horizontal stacks of border tiles. The equivalent particle configuration in TASEP is therefore its initial megajam with all particles (bullets) stacked vertically and holes (circles) stacked horizontally as shown in part (a) of Figure 2.3.

Note that the time, \( C_{M,N} \) to place a tile at \((M, N)\) is equal in distribution to the time that, under the TASEP, the \(M\)-th hole exchanges its position with the \(N\)-th particle, where particles are enumerated bottom-up and holes are enumerated left-to-right as shown in Figure 2.3. For example, when the particle-hole exchange (tile attachment) shown in Figure 2.3 takes place, the rectangle with vertices at \((1,1)\) and \((3,6)\) has been fully assembled. Thus, the analysis of rectangle self assembly reduces to precisely the analysis of TASEP passage times described in Chapter III.1.

2.1.2 Asymptotics of \( C_{M,N} \)

Since the random variable \( C_{M,N} \) defined for rectangle self assembly has the same distribution as the \( C_{M,N} \) defined for the TASEP, we now confine ourselves to the TASEP process and its two-dimensional representations as in Figure 2.3. Recall the asymptotic result for TASEP given in Chapter 1:

The hydrodynamic limit of \( C_{M,N} \) is defined as

\[
C_{x,y} := \lim_{n \to \infty} \frac{1}{n} C_{\lfloor nx \rfloor, \lfloor ny \rfloor}
\]

(2.3)

By varying \( C_{x,y} \), we obtain the limiting shape of the profile as a function of time. Figure 2.4 illustrates these shapes where time increases as the curves move outward along the diagonal. These shapes should not be surprising. In the discrete system,
the first row and column always have an available site, and so the rate of attachments along this row and column is maximum at 1. Progressively, the rate of attachment decreases as the sites where rows and columns intersect become further removed from the boundaries and closer to the diagonal, where the rate of tile attachments is smallest. Below, we expand further on this behavior, which is clearly symmetric about the diagonal.

![Graph showing expected shapes of a tile self-assembly crystal with the L-shape border in the hydrodynamic limit. The figure shows the shapes of the self-assembly for different times, the smallest time provides the smallest profile. The x-axis and y-axis represent the borders.](image)

Figure 2.4: Expected shapes of a tile self-assembly crystal with the L-shape border in the hydrodynamic limit. The figure shows the shapes of the self-assembly for different times, the smallest time provides the smallest profile. The x-axis and y-axis represent the borders.

Let $Y_{i,j}$ be the number of paths $\mathcal{P}$ from $(1, 1)$ to $(i, j)$; beyond the initial tile, each tile in such a path is either above or to the right of the previous tile in the path. Among these paths, let $\mathcal{P}_{\text{max}}(i, j)$ denote the one called the max-path which has the largest duration, defined as the sum of the times that available sites along the path spent waiting for a tile to bond. All paths counted by $Y_{i,j}$ have length $i + j - 1$. No pair of such paths is statistically independent, but the marginal durations of all such paths are distributed as the sum of $i + j - 1$ rate-1 exponentials. On the other hand, $Y_{i,j}$ depends on $i, j$, and therefore, by (2.1), the likelihood of $\mathcal{P}_{\text{max}}(i, j)$ passing
CHAPTER 2. COMPUTING TIMES

through \((i - 1, j)\) will differ from the likelihood of its passing through \((i, j - 1)\) (the maximum over the larger set of paths will be larger stochastically than it is over the smaller set). As a simple example to fix ideas, take \(M = 2, N = 3\), so that \(Y_{2,3} = 3\), \(Y_{1,3} = 1\), \(Y_{2,2} = 2\); in computing \(C_{2,3}\) we find that

\[
C_{1,3} = T_{1,1} + T_{1,2} + T_{1,3}, \quad C_{2,2} = \max\{T_{1,1} + T_{1,2} + T_{2,2}, T_{1,1} + T_{2,1} + T_{2,2}\}
\]

with the latter clearly being stochastically larger than the former.

More precisely, we have

\[
Y_{M,N} = \binom{M+N-2}{M-1} = \binom{M+N-2}{N-1}, \quad M, N \geq 1
\]  

(2.4)

with \(Y_{M,1} = Y_{1,N} = 1\), which is easily proved by induction on path length, using the recurrence \(Y_{M,N} = Y_{M-1,N} + Y_{M,N-1}\), or by the simple observation that there are \(\binom{M+N-2}{M-1}\) ways of choosing the \(M - 1\) moves a path must make to the right, which is the same as the number \(\binom{M+N-2}{N-1}\) of ways of choosing the \(N - 1\) moves a path must make upward.

As illustrated in the above example, \(M < N\) implies that \(\binom{M+N-3}{M-2} < \binom{M+N-3}{M-1}\) (i.e., the number of paths from (1,1) to \((M, N - 1)\) is greater than the number from (1,1) to \((M - 1, N)\), and so when \(M < N\), the max-path \(P_{\text{max}}(M, N)\) is more likely to pass through \((M, N - 1)\) than it is to pass through \((M - 1, N)\). Thus, starting at \((M, N)\) and working back towards (1,1), the max-path has a greater tendency to move down than to the left, the magnitude of the trend being determined by the ratio \(M/N\).

In the continuous, hydrodynamic limit the discrete max-paths become straight lines, the diagonals with slopes \(y/x\) from \((0,0)\) to \((x,y)\) where \(x, y\) are the continuous limits of \(M, N\) under the hydrodynamic scaling. A symmetric argument exchanging \(M\) and \(N\), \(x\) and \(y\), and the left border with the lower border applies to the case \(M > N\).
CHAPTER 2. COMPUTING TIMES

Figure 2.5 shows the straight-line, hydrodynamic limit of $\mathcal{P}_{\text{max}}(M, N)$. In this chapter, we show that the profile obtained from the hydrodynamic limit is in very close agreement with the profile obtained from experiments. This also is as expected, considering the scale of molecular processes.

![Graph showing the path $\mathcal{P}_{\text{max}}(M, N)$ in the hydrodynamic limit.]

Figure 2.5: The path $\mathcal{P}_{\text{max}}(M, N)$ in the hydrodynamic limit.

2.2 Growing Borders

In Section 2.1, our work was confined to a growth process in which borders did not participate. But in the laboratory, experiments in which borders grow in parallel with the rule-tile structures are also common. This would not produce an interesting generalization to our mathematical model were it not for the fact that the rate of border-tile attachment, say $\alpha$, is normally slower than the rule-tile rate. For, with the slower border growth rate, new classes of crystal shapes are achievable in our self assembly model, as will be seen below.\(^1\)

\(^1\)K. Fujibayashi working in E. Winfree's lab at the California Institute of Technology notes that an $\alpha$ approximately 1/2 of the rule-tile rate was common in certain of his experiments, where $\alpha$ is a stoichiometric parameter giving the concentration of border tiles. In this case he found that the profile of the self-assembly structure is triangular, a fact that will also emerge in our purely
CHAPTER 2. COMPUTING TIMES

With borders also growing, the full self assembly process now includes a column 0 and a row 0 representing the vertical and horizontal sequences of border tiles; \( T_{0,j} \) and \( T_{i,0} \) now denote the tile waiting times along the vertical and horizontal borders. It is convenient to keep the rule-tile rate parameter normalized to 1, and to let \( \alpha \) denote the rate parameter for border-tile growth. Bear in mind that both boundaries grow at this rate and that an exponential distribution also applies to the border tile waiting times, \( T_{0,j} \), \( T_{i,0} \).

As defined earlier in (2.3), the hydrodynamic limit of \( C_{M,N} \) is

\[
C_{x,y} = \lim_{n\to\infty} \frac{1}{n} C_{[nx],[ny]}
\]

A formula for \( C_{x,y} \) can be developed informally from an analysis in the original discrete model of the max-paths \( \mathcal{P}_{\text{max}}(M, N) \) as a function of \( \alpha \) as well as \( M, N \). To this end, we restrict the discussion to the case \( M < N \); a symmetric argument applies when \( M > N \). Since border assembly has now been made a part of the self assembly process, max-paths now pass through border tiles in general. There is an up-right constraint for path segments away from the borders; and once a max-path enters the border tiles, as one proceeds along the path, say, from \( M, N \) down to \( (0, 0) \), the path must stay there.

Consider first the extremes of \( \alpha \). Clearly, since \( M < N \), then as \( \alpha \to 0 \) and hence as borders take very long to grow, the max-paths will tend to be entirely in the left (vertical) border from 0 to \( N \) before crossing over to \( (M, N) \); the max-path contains the line segment from \( (0, 0) \) to \( (0, y) \) in the hydrodynamic limit. On the other hand, it is not difficult to verify that the border has no effect on max-paths when \( \alpha > 1 \); the hydrodynamic limit of max-paths will never include any part of the border in this case. Instead, they will coincide with the diagonal paths of Figure 2.5 in this limit. mathematical framework.
In the general case $0 < \alpha < 1$, one encounters in the hydrodynamic limit either  
(a) simply straight lines (rays from the origin) when $x$ is far enough from the border and $\alpha$ is large enough, or  
(b) a two-stage path consisting first of a border segment starting at the origin, and then a straight line starting at the end of the border segment and ending at $(x, y)$.  

These two cases are illustrated in Figure 2.6 along with a case where $x > y$.  

To explain this limiting behavior within the discrete model, the obvious, first fact to note is that the larger mean of the waiting times $T_{0,j}, T_{i,0}$ biases max-paths toward the border. When a segment along a border is included in a max-path to $(M, N)$, it means that the durations of max-paths to $(M - 1, N)$ with segments in the border dominate those of max-paths to $(M, N - 1)$, even though the latter are computed as maxima over larger path sets, i.e., even though $Y_{M,N-1} > Y_{M-1,N}$. But in the hydrodynamic limit, if $\alpha < 1$ is large enough, and $x$ is far enough from the border ($y$-axis), this duration-dominance of paths with segments in the borders must yield to the greater durations of max-paths computed as maxima over larger path sets.  

Finding the articulation point dividing the two possible segments in the hydrodynamic limit is a simple calculus problem. Along a border segment of length $d$, the limiting sum of waiting times becomes simply $d/\alpha$, but along a diagonal from $(0, y - z)$ to $(x, y)$ the limiting sum is given by (2.3), viz., $(\sqrt{x} + \sqrt{z})^2$. It follows that  

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

(2.5)  

Setting the derivative with respect to $z$ equal to 0 yields  

$$2(\sqrt{x} + \sqrt{z}) \cdot \frac{1}{2\sqrt{z}} - \frac{1}{\alpha} = 0$$
so after checking that the second derivative is negative, one finds that the value $z_*$ maximizing (2.5) is given by

$$z_* = \left( \frac{\alpha}{1 - \alpha} \right)^2 x$$  \hspace{1cm} (2.6)

provided $z_* < y$. Now, $x < y$ and $z_* < y$ imply that $(\alpha/(1 - \alpha))^2 > 1$, or, $\alpha > 1/2$, so for $\alpha < 1/2$, the max-paths consist of a border segment and a diagonal segment, where the two are joined at $(0, y - z_*)$ with $z_*$ given by (2.6). A similar max-path is obtained for $\alpha > 1/2$, unless $x$ is sufficiently near the border, in which case $z_* = y$ and the max-paths in the hydrodynamic limit are simple diagonals (cf. Figure 2.6 for examples).

![Figure 2.6: The paths that contribute to the time required for a tile to attach to the positions $(x_i, y_i)$. When $(x_i, y_i)$ is far enough from the border and $\alpha$ is large enough, the max-path is a ray from the origin. Otherwise, the max-path is a two-stage path consisting of a border segment starting at the origin and then a straight line starting at the end of the border segment and ending at $(x_i, y_i)$.](image)

The new classes of profiles/shapes are illustrated in Figure 2.7 with $\alpha$ a parameter and the time taken to be $C_{x,y} = 1000$. As shown, when $\alpha < 1/2$, the profiles are symmetric about the diagonal and linear on both sides of the diagonal. The special
case of $\alpha = 1/2$ gives the triangular structure, which we mentioned earlier is consistent with laboratory experiments. The equations for these cases are obtained by substituting (2.6) into (2.5). A little algebra yields

**Case 1: $0 < \alpha \leq 1/2$**

$$C_{x,y}(\alpha) = \begin{cases} \frac{x}{1-\alpha} + \frac{y}{\alpha} & x < y, \\ \frac{x}{\alpha} + \frac{y}{1-\alpha} & x \geq y \end{cases} \quad (2.7)$$

For $1/2 < \alpha < 1$, the linear segments near the border terminate at the points where the time $C_{x,y}$ obtained from (2.7) starts to be less than that of (2.3) which is when $z_\ast = y$. After a little algebra, we have

**Case 2: $1/2 \leq \alpha < 1$**

$$C_{x,y} = \begin{cases} \frac{x}{1-\alpha} + \frac{y}{\alpha} & x < \left(\frac{1-\alpha}{\alpha}\right)^2 y, \\ \left(\sqrt{x} + \sqrt{y}\right)^2 \left(\frac{1-\alpha}{\alpha}\right)^2 y < x < \left(\frac{\alpha}{1-\alpha}\right)^2 y, \\ \frac{y}{1-\alpha} + \frac{x}{\alpha} & x > \left(\frac{\alpha}{1-\alpha}\right)^2 y \end{cases} \quad (2.8)$$

The diagonal max-path in Figure 2.7 illustrates the case when $\left(\frac{1-\alpha}{\alpha}\right)^2 y < x < \left(\frac{\alpha}{1-\alpha}\right)^2 y$. In this region, Equation (2.6) yields $z_\ast > y$ which contradicts our original assumption. Thus, the value of $z_\ast$ must be $y$ and the profile in this region follows a curve $(\sqrt{x} + \sqrt{y})^2 = \text{const.}$

### 2.3 Experiments

To test the usefulness of our fluid-limit results, we conducted a number of experiments using straightforward simulation techniques. A conclusion worth emphasizing is the very close agreement that we shall find between the experimental and analytical results. But before presenting the simulation data, we briefly outline the design of the simulation program (details of the algorithm can be found in the appendices).
2.3.1 Algorithmics

Let \( t_k, \ k = 0, 1, \ldots \) be the epochs of a simulation of the Markov tile self-assembly process, i.e., the times at which tiles are attached to the crystal, with \( t_0 = 0 \) and the state at time 0 indicating that only the seed tile is present. Consider the more general model in which, during the interval \( \delta_k := [t_k, t_{k+1}) \), the growth along the vertical border is at rate \( \alpha_k \) and that along the horizontal border is \( \beta_k \). The state is given by a nonincreasing sequence of positive row lengths \( (r_k(0), r_k(1), \ldots) \). See the example in Figure 2.8. The initial state is then the single integer \( r_0(0) = 1 \) which just counts the seed in the first row. The durations of the intervals \( \delta_k \) are independent samples from exponential distributions with rate parameters \( \alpha_k + \beta_k + C_k \), where \( C_k \geq 0 \) is the number of available sites (corners) in the state during \( \delta_k \), not counting those at the boundaries. Clearly, such corners appear wherever \( r_k(i) > r_k(i + 1) \).

Recall that at all such sites i.i.d. rate-1 exponential random variables model the waiting times until tiles bond to the sites. Note that, if the interest is only in the self assembly of an \( M \times N \) rectangle, then (i) the profile-state information can be restricted to the rectangle with vertices at the origin and position \( (M, N) \), and (ii) \( \alpha_k \) can be set to 0 when the number of nonempty rows reaches \( N \) and \( \beta_k \) can be set to 0 if \( r_k(0) \geq M \), i.e. border growth rates beyond the rectangle can be set to 0. \( C_0 \) starts at 0, and remains nonnegative until time \( C_{M,N} \), when the rectangle is completely assembled and \( C_\ell = 0 \), with \( \ell = C(M, N) \). At each epoch \( t_k, \ k > 0 \), a sample from the distribution

\[
\left\{ \frac{\alpha_k}{\alpha_k + \beta_k + C_k}, \frac{\beta_k}{\alpha_k + \beta_k + C_k}, \frac{C_k}{\alpha_k + \beta_k + C_k} \right\}
\]

determines whether a tile has attached at the vertical border, the horizontal border, or one of the interior sites for rule tiles. If a rule-tile attachment has occurred, its location is determined by a sample from the uniform distribution on the \( C_k \) sites available to rule tiles.
2.3.2 Comparisons

Let the border growth rate be symmetric and constant with $\alpha_k = \beta_\ell = \alpha$ for all $k, \ell$ such that $\alpha_k, \beta_\ell > 0$. Figure 2.9 was produced by Fujibayashi from kinetic simulation data based on Winfree’s kinetic Tile Self-Assembly Model. The figure shows the intermediate shapes of self-assembly structures for various values of $\alpha$.

For comparison, in Figure 2.10, our simulation results are plotted along with the hydrodynamic limit results obtained from (2.7) and (2.8) for different values of $\alpha$. The simulation was performed for one million tile attachments. It is obvious from Figure 2.10 that, for all $\alpha$, the simulation data agree remarkably well with our hydrodynamic-limit estimates of the shapes of self-assembled structures.
Figure 2.7: Profiles at time 1000. For $0 < \alpha \leq 0.5$, when the effect of the border attachment rate is strong, the shape of the structure is convex; at the threshold $\alpha = 0.5$, the shape of the structure is a triangle. When $\alpha > 0.5$, the effect of the border growth rate is weak and the profile is concave; it is linear near the border of the growth, but follows a curve $(\sqrt{x} + \sqrt{y})^2 = const$ in the middle. For $0 < \alpha \leq 0.5$, the linear line representing the boundary of the profile is $\alpha(1000-x/(1-\alpha))$ for $x \leq y$. For $0.5 \leq \alpha < 1$, the linear line near the border is described by $\alpha(1000-x/(1-\alpha))$ for $0 < x \leq (\frac{1-\alpha}{\alpha})^2 y$, and the curve in the middle of the profile is described by $(\sqrt{1000}-\sqrt{x})^2$ for $(\frac{1-\alpha}{\alpha})^2 y \leq x < 1$. For $\alpha \geq 1$, the curve is described by $(\sqrt{1000}-\sqrt{x})^2$. Since the profile is symmetric along $x = y$, the linear line and the curve describing the profile can be obtained from the case where $x \leq y$. 
Figure 2.8: The state of the self-assembly at time $t_k$ is maintained by the array $r_k(i)$, $i \geq 0$, where $t_k$, $k = 0, 1, \ldots$, are the times at which tiles are attached to the crystal, with $t_0 = 0$. The array $r_k(i)$ keeps track of the non-increasing row lengths in the self-assembly profile. In this example, the time index is the total number of grid squares, $k = 39$. 

$\begin{align*}
r_k(0) & = 9 \\
r_k(1) & = 7 \\
r_k(2) & = 6 \\
r_k(3) & = 6 \\
r_k(4) & = 5 \\
r_k(5) & = 3 \\
r_k(6) & = 3
\end{align*}$
Figure 2.9: Simulation of 2D structures with various values of $\alpha$. 
CHAPTER 2. COMPUTING TIMES

Figure 2.10: Simulation of 2D structures with various values of $\alpha$. 
Chapter 3

Parallelism Performance

This chapter studies the performance of tile self-assembly in terms of the parallelism in the tile attachment process. In the unconstrained, or open-ended, process of tiling the northeast quadrant, if $r(t)$ is the number of available sites in the profile at time $t$, then by our exponential assumption on attachment times, the time to the next attachment is the minimum of rate-1 exponentials - another exponential - with rate parameter $r(t)$. Thus, $r(t)$ is a natural measure of parallelism. As before, an exact analysis of $r(t)$ appears to be very difficult, and so we resort to asymptotic methods, in particular hydrodynamic limits. On the other hand, the positive side of this situation bears repeating: at molecular scaling very little accuracy is lost in the hydrodynamic limit.

In the hydrodynamic limit, $r(t)$ becomes the rate of change of the continuous tile mass under the profile; the limiting tile mass is denoted by $\tilde{m}(t)$ and its derivative is denoted by $\tilde{r}(t) = \frac{d}{dt}\tilde{m}(t)$. We begin with the easier analysis of open-ended self-assembly, i.e., the unbounded process of self-assembly in the positive lattice. We then turn to the calculation of $\tilde{r}(t)$ in rectangle self-assembly; the methods still involve only elementary calculus, but the calculations become rather tedious. Note that, for rectangle self-assembly, it is only the rate of change of the tile mass inside the rectangle.
that determines the measure of parallelism.

One expects that, for the open-ended self-assembly process, the degree of parallelism should increase linearly in time, since the shapes of the profiles are the same throughout the self-assembly process. For rectangle self-assembly, the degree of parallelism should increase initially and then decrease to zero as the assembly of the rectangle nears completion at the northeast corner. These expectations are borne out by the results below.

3.1 Open-Ended Self-Assembly

For open-ended self-assembly, the limiting tile mass is the area under the self-assembly profile. Figure 3.1 illustrates the area under the profile for self-assembly processes with different values of the common border growth-rate parameter \( \alpha \).

**Case 1: \( 0 \leq \alpha \leq 0.5 \)**

The area under the profile consists of the area of the trapezoid \( \{ (0, 0), (x_0, 0), (x_0, y_0), (0, \alpha t) \} \) and the area of the triangle \( \{ (x_0, y_0), (0, \alpha t), (x_0, 0) \} \) (see the shaded area in Figure 3.1(a)). We then have

\[
\tilde{m}(t) = \frac{1}{2}(\alpha t + y_0)x_0 + \frac{1}{2}(\alpha t - x_0)y_0
\]

\[
= \alpha^2(1 - \alpha)t^2
\]

where \( (x_0, y_0) \) is the point where the lines defining the profile cross and

\[ x_0 = y_0 = \alpha(1 - \alpha)t. \]
CHAPTER 3. PARALLELISM PERFORMANCE

Figure 3.1: The area under the profile curves represents the tile-mass in the self-assembly structure $\tilde{N}(t)$. This figure shows the profiles with different $\alpha$’s.

\textbf{CASE 2: $0.5 \leq \alpha \leq 1$}

The area under the profile can be separated into three regions determined by where the linear characteristic of the profile turns into the curve $\sqrt{t} = \sqrt{x} + \sqrt{y}$ and vice versa. Let the points where the linear profile changes into the curved profile be $(x_*, y_*)$ and $(x_{**}, y_{**})$ as shown in Figure 3.1(b). This area consists of three different regions: the trapezoid $\{(0, 0), (0, x_*), (x_*, y_*), (0, \alpha t)\}$, the inner curved profile and
Lastly the triangle \{ (at, 0), (x^{**}, y^{**}), (x^{**}, 0) \}. Thus, we have

\[
\tilde{m}(t) = \frac{1}{2} (at + y_*) x_* + \int_{x_*}^{x^{**}} (\sqrt{t} - \sqrt{x})^2 dx + \frac{1}{2} (at - x^{**}) y^{**}
\]
\[
= \frac{t^2}{6} (-1 + 6\alpha - 6\alpha^2 + 2\alpha^3)
\]

where

\[
x_* = y^{**} = (1 - \alpha)^2 t
\]
\[
y_* = x^{**} = \alpha^2 t
\]

**CASE 3: \( \alpha \geq 1 \)**

When \( \alpha \geq 1 \), the profile is defined by the curve \( y = (\sqrt{t} - \sqrt{x})^2 \). The area under the profile is the integral of this function from \( x = 0 \) to \( x = at \). This gives us

\[
\tilde{m}(t) = \int_0^{at} (\sqrt{t} - \sqrt{x})^2 dx
\]
\[
= \frac{t^2}{6} (6\alpha - 8\alpha^{3/2} + 3\alpha^2)
\]

We can see that the tile mass \( \tilde{m}(t) \) is a quadratic function of \( t \). Thus, the derivative, the rate of rule-tile attachment, is linear in \( t \), as follows.

\[
\tilde{r}(t) = \begin{cases} 
2\alpha^2(1 - \alpha)t, & 0 \leq \alpha \leq 0.5 \\
[1 - 2(1 - \alpha)^3]t/3, & 0.5 \leq \alpha \leq 1 \\
[2 + (3\sqrt{\alpha} - 2)(\sqrt{\alpha} - 2)at/3, & \alpha \geq 1
\end{cases}
\]

This makes sense because the profile shape of the self-assembly process stays the same for each value of \( \alpha \). For the discrete self-assembly process, the linear rate of rule-tile attachment can be interpreted as a linear growth rate in the number of available sites for rule tiles.
3.2 Rectangle Self-Assembly

For this case, we are interested in the parallelism over the time taken to construct an \( a \times b \) rectangle where \( a, b \in \mathbb{R} \). Bear in mind that we are still arguing in terms of the hydrodynamic limit. The tile mass of interest is now represented by the area under the profile curve and inside the \( a \times b \) rectangle. The calculations become awkward because the boundary of the area \( \tilde{m}(t) \) varies as self assembly progresses. The case analysis is worked out in detail, but the calculations are tedious so we take opportunities to omit routine details. Without loss of generality, we assume that \( b \leq a \). Figure 3.2 shows the different cases that have to be considered for the calculation of \( \tilde{m}(t) \) for each \( \alpha \). For \( a \geq b \), the result would be the same with \( a \) and \( b \) interchanged.

**Case 1:** \( 0 < \alpha \leq 0.5 \)

As shown in Figure 3.2(a), the intersection of the rectangle and the area under the profile can be broken down into 5 different cases. According to the figure, we define important boundary values as follows:

- \((x_*, y_*)\) is the point where both lines that describe the profile intersect (see Equation (2.7)).
- \((x', b)\) is the point where the line describing the self-assembly profile \( t = x/(1 - \alpha) + y/\alpha \) intersects the top border of the \( a \times b \) rectangle.
- \((x'', b)\) is the point where the line describing the self-assembly profile \( t = x/\alpha + y/(1 - \alpha) \) intersects the top border of the \( a \times b \) rectangle.
- \((a, y'')\) is the point where the line describing the self-assembly profile \( t = x/\alpha + y/(1 - \alpha) \) intersects the right border of the \( a \times b \) rectangle.

where
Figure 3.2: This figure shows the evolution of the profiles for different $\alpha$'s.

\[ x_* = \alpha(1 - \alpha)t, \quad y_* = \alpha(1 - \alpha)t, \]
\[ x' = (1 - \alpha) \left( t - \frac{b}{\alpha} \right), \quad x'' = \alpha \left( t - \frac{b}{1 - \alpha} \right), \]
\[ y''' = (1 - \alpha) \left( t - \frac{b}{\alpha} \right) \]

Elementary calculus yields the following results.

**CASE 1.1: $t \leq b/\alpha$**

In this case, $\bar{m}(t)$, is the area under the profile described by line (1) of Fig-
ure 3.2(a) and inside the a × b rectangle; \( \tilde{m}(t) \) is twice the area of the triangle \{(0, 0), (x_*, y_*), (0, \alpha t)\}, so

\[
\tilde{m}(t) = 2 \left( \frac{1}{2} \alpha t x_* \right)
= \alpha t x_*
\]

CASE 1.2: \( b/\alpha \leq t < a/\alpha \) and \( x_* \leq b \)

The tile mass, \( \tilde{m}(t) \), is now the area under the profile described by line (2) of Figure 3.2(a) and inside the rectangle; it contains the rectangle \{(0, 0), (x', 0), (x', b), (0, b)\}, the trapezoid \{(x', 0), (x_*, 0), (x_*, y_*), (x', b)\}, and the triangle \{(x_*, 0), (\alpha t, 0), (x_*, y_*)\}. Thus,

\[
\tilde{m}(t) = bx' + \frac{1}{2}(b + y_*)(x_* - x') + \frac{1}{2}(\alpha t - x_*)y_*
\]

CASE 1.3: \( b/\alpha \leq t < a/\alpha \) and \( x_* \geq b \)

In this simple case, \( \tilde{m}(t) \), is the area under the profile described by line (3) of Figure 3.2(a) and inside the a × b rectangle; \( \tilde{m}(t) \) contains the rectangle \{(0, 0), (x'', 0), (x'', b), (0, b)\}, and the triangle \{(x'', 0), (\alpha t, 0), (x'', b)\}. Thus,

\[
\tilde{m}(t) = bx'' + \frac{1}{2}(\alpha t - x'')b
\]

CASE 1.4: \( \frac{a}{\alpha} \leq t \leq \frac{a}{\alpha} + \frac{b}{1-\alpha} \)

We see that \( \tilde{m}(t) \) is now the area under the profile described by line (4) of Figure 3.2(a) and inside the a × b rectangle; \( \tilde{m}(t) \) contains the rectangle \{(0, 0), (x'', 0), (x'', b), (0, b)\}, and the trapezoid \{(x'', 0), (a, 0), (a, y''), (x'', b)\}. Thus,

\[
\tilde{m}(t) = bx'' + \frac{1}{2}(b + y'')(a - x'')
\]
CHAPTER 3. PARALLELISM PERFORMANCE

Case 1.5: $t \geq \frac{a}{c} + \frac{b}{1-c}$

In this final case, the $a \times b$ rectangle is completely filled, so that

$$\tilde{m}(t) = ab$$

Differentiate $\tilde{m}(t)$ with respect to $t$ for each case, and obtain $\tilde{r}(t)$.

Case 2: $0.5 \leq \alpha \leq 1$

According to Figure 3.2(b), we define boundary values as follows:

- $(x_*, y_*)$ and $(x_{**}, y_{**})$ are the points where the linear profile changes to the curved profile and vice versa with $x_* \leq x_{**}$ and $y_* \geq y_{**}$.

- $(x', y')$ is the point where the top linear part of the profile intersects the top boundary of the $a \times b$ rectangle.

- $(x', y')$ is the point where the bottom linear part of the profile intersects the right boundary of the $a \times b$ rectangle.

- $(x'', y'')$ is the point where the curved part in the middle of the profile intersects the top boundary of the $a \times b$ rectangle.

where

$$
x_* = (1 - \alpha)^2 t, \quad y_* = \alpha^2 t, \\
x_{**} = \alpha^2 t, \quad y_{**} = (1 - \alpha)^2 t, \\
x' = (1 - \alpha) \left( t - \frac{b}{\alpha} \right), \quad y' = (1 - \alpha) \left( t - \frac{a}{\alpha} \right), \\
x'' = (\sqrt{t} - \sqrt{b})^2, \quad y'' = (\sqrt{t} - \sqrt{a})^2,$$
CHAPTER 3. PARALLELISM PERFORMANCE

The case analysis for \( \widetilde{m}(t) \) is as follows:

**CASE 2.1:** \( t \leq b/\alpha \)

Referring to (2.8), \( \widetilde{m}(t) \) is the area under the profile described by line (1) of Figure 3.2(b) and inside the \( a \times b \) rectangle, so \( \widetilde{m}(t) \) consists of the area of the trapezoid \( \{(0,0), (x_*,0), (x_*,y_*), (0, \alpha t)\} \), the area under the curve \( y = (\sqrt{t} - \sqrt{x})^2 \) from \( x = x_* \) to \( x = x_{**} \), and the area of the triangle \( \{(x_{**},0), (\alpha t,0), (x_{**},y_{**})\} \) (see profile (1) of Figure 3.2(b)). Then,

\[
\widetilde{m}(t) = \frac{1}{2}(\alpha t + y_*)x_* + \int_{x_*}^{x_{**}} (\sqrt{t} - \sqrt{x})^2 dx + \frac{1}{2}(\alpha t - x_{**})y_{**}
\]

**CASE 2.2:** \( b/\alpha \leq t < a\alpha \) and \( y_* \leq b \)

The inequalities of this case imply, from (2.8), that \( \widetilde{m}(t) \) is the area under the profile described by line (2) of Figure 3.2(a) and inside the rectangle, so \( \widetilde{m}(t) \) consists of the area of the rectangle \( \{(0,0), (x',0), (x',b), (0,b)\} \), the area of the triangle \( \{(x',y_*), (x_*,y_*), (x',b)\} \), the area under the curve \( x = (\sqrt{t} - \sqrt{y})^2 \) from \( y = y_* \) to \( y = y_{**} \), and the area of the trapezoid \( \{(0,0), (\alpha t,0), (x_{**},y_{**}), (0,y_{**})\} \) (see profile (2) of Figure 3.2(b)). Then,

\[
\widetilde{m}(t) = x'b + \frac{1}{2}(x_* - x')(b - y_*) + \int_{y_*}^{y_{**}} (\sqrt{t} - \sqrt{y})^2 dy + \frac{1}{2}(x_{**} + \alpha t)y_{**}
\]

**CASE 2.3:** \( b/\alpha \leq t < a/\alpha \) and \( y_* \geq b \)

Equation (2.8) in this case shows that \( \widetilde{m}(t) \) is the area under the profile described by line (3) of Figure 3.2(a) and inside the rectangle; \( \widetilde{m}(t) \) consists of the area of the rectangle \( \{(0,0), (x'',0), (x'',b), (0,b)\} \), the area under the curve \( y = (\sqrt{t} - \sqrt{x})^2 \) from \( x = x'' \) to \( x = x_{**} \), and the area of the triangle \( \{(x_{**},0), (\alpha t,0), (x_{**},y_{**})\} \) (see profile (3) of Figure 3.2(b)). Thus,
\[ \tilde{m}(t) = bx''' + \int_{x''}^{x'''} (\sqrt{t} - \sqrt{x})^2 dx + \frac{1}{2} (\alpha t - x_{**}) y_{**} \]

**Case 2.4:** \( t > a/\alpha, x_{**} \leq a, \) and \( y_* \geq b \)

Here, \( \tilde{m}(t) \) is the area under the profile described by line (4) of Figure 3.2(a) and inside the rectangle; it consists of the area of the rectangle \( \{(0,0), (x''',0), (x'''', b), (0,b)\} \), the area under the curve \( y = (\sqrt{t} - \sqrt{x})^2 \) from \( x = x''' \) to \( x = x_{**} \), and the area of the trapezoid \( \{(x_{**},0), (a,0), (a,y''), (x_{**},y_{**})\} \) (see profile (4) of Figure 3.2(b)). Hence,

\[ \tilde{m}(t) = bx''' + \int_{x''}^{x''''} (\sqrt{t} - \sqrt{x})^2 dx + \frac{1}{2} (y_{**} + y'')(a - x_{**}) \]

**Case 2.5:** \( a/\alpha < t \leq (\sqrt{a} + \sqrt{y})^2, x_{**} \geq a, y_* \geq b \)

The tile mass \( \tilde{m}(t) \) is the area under the profile described by line (4) of Figure 3.2(a) and inside the rectangle; it consists of the area of the rectangle \( \{(0,0), (x''',0), (x'''', b), (0,b)\} \), and the area under the curve \( y = (\sqrt{t} - \sqrt{x})^2 \) from \( x = x''' \) to \( x = a \) (see profile (5) of Figure 3.2(b)). Thus,

\[ \tilde{m}(t) = bx''' + \int_{x''}^{a} (\sqrt{t} - \sqrt{x})^2 dx \]

**Case 2.6:** \( t \geq (\sqrt{a} + \sqrt{y})^2 \)

In this case, the \( a \times b \) rectangle is completely filled and we have

\[ \tilde{m}(t) = ab \]

An explicit expression for \( \tilde{r}(t) \) covering the various cases can be found by a routine differentiation which we omit.
CHAPTER 3. PARALLELISM PERFORMANCE

CASE 3: $\alpha \geq 1$

According to Figure 3.2(c), the tile mass $\tilde{m}(t)$, is the area under the profile described by Figure 3.2(c) and inside the rectangle. We have

$$
\tilde{m}(t) = \begin{cases} 
\int_0^t (\sqrt{t} - \sqrt{x})^2 \, dx, & t \leq b \\
\int_0^b (\sqrt{t} - \sqrt{y})^2 \, dy, & b \leq t \leq a \\
bx + \int_x^a (\sqrt{t} - \sqrt{x})^2 \, dx, & a \leq t \leq (\sqrt{a} + \sqrt{b})^2 \\
ab, & t \geq (\sqrt{a} + \sqrt{b})^2
\end{cases}
$$

where $(x_*, b)$ is the point where the curve $y = (\sqrt{t} - \sqrt{x})^2$ intersects the line $y = b$, and $x_* = (\sqrt{t} - \sqrt{b})^2$. This gives us

$$
\bar{r}(t) = \begin{cases} 
\frac{t}{3}, & t \leq b \\
b(1 - \frac{2}{3}\sqrt{b/t}), & b \leq t \leq a \\
-\frac{t}{3} + (a + b) - \frac{2}{3}(\sqrt{a^3/t} + \sqrt{b^3/t}), & a \leq t \leq (\sqrt{a} + \sqrt{b})^2 \\
0, & t \geq (\sqrt{a} + \sqrt{b})^2
\end{cases}
$$

Figure 3.3 and 3.4 show the evolution of $\tilde{m}(t)$ and $\bar{r}(t)$, respectively, for a 1200×700 rectangle, and is as we expected. The tile mass $\tilde{m}(t)$ increases quickly in the beginning, then increases slowly and stays at $ab$. Thus, the rate of parallelism increases at the beginning and then drops down to zero as the time reaches $C_{ab}$.

Note the distinctive behavior of $\bar{r}(t)$ in Figure 3.4, for $\alpha \leq 0.5$. This is because the profile of the self-assembly structure for $\alpha \leq 0.5$ has a simple, piece-wise linear geometric shape. The most interesting subcase occurs when $\bar{r}(t)$ stays constant with $b \leq \alpha t < a$ and $\alpha(1 - \alpha)t \geq b$ (see Section 3.2). According to Figure 3.4(a), when $b \leq \alpha t < a$ and $\alpha(1 - \alpha)t \geq b$, the area describing $\tilde{m}(t)$ is a trapezoid (see profile (3) of Figure 3.4(a)) with
\[ \tilde{m}(t) = bx'' + \frac{1}{2}(\alpha t - x'')b \]
\[ = bat - \frac{b^2 \alpha}{2(1 - \alpha)} \]

where

\[ x'' = \alpha \left( t - \frac{b}{1 - \alpha} \right) \]

Since \( \tilde{m}(t) \) is linear in \( t \) when \( b \leq \alpha t < a \) and \( \alpha(1 - \alpha)t \geq b \), then \( \tilde{r}(t) \) is a constant in this time interval, as shown in Figure 3.4(a).

### 3.3 Experiments

We counted the number of rule-tile attachments as a function of time in a simulation of open ended self-assembly. Our findings are illustrated in Figure 3.5. As can be seen, the experimental results agree closely with the results of our analysis of the hydrodynamic limit.

For the case of rectangle self assembly, we again chose a 1200 × 700 rectangle. Figure 3.6 shows the number of tiles in the rectangle as a function of time for different values of the border growth rate \( \alpha \). The tile mass from our calculation in the preceding section again fits the number of tiles from the simulation very well.
Figure 3.3: This figure shows the evolution of $\hat{m}(t)$ for different $\alpha$'s for the rectangle self-assembly. The self-assembly is completed when the rectangle of size $1200 \times 700$ is constructed.
Figure 3.4: This figure shows the evolution of $\hat{r}(t)$ for different $\alpha$’s for the rectangle self-assembly. The self-assembly is completed when the rectangle of size $1200 \times 700$ is constructed.
Figure 3.5: The number of rule tiles with respect to time for various values of $\alpha$ for the open-ended self-assembly. The plot shows the number of rule tiles from our simulation and our calculation from Chapter 3.
Figure 3.6: The number of rule tiles with respect to time for various values of \( \alpha \) for the rectangle self-assembly. The plot shows the number of rule tiles from our simulation and our calculation from Chapter 3.
Chapter 4

Error-Correction

4.1 Introduction

Previously, we explored the speed of tile self-assembly in a perfect world where every tile that attaches to a candidate site is the correct tile, according to the design of the self-assembly system, and every such tile attaches correctly. But errors in both senses can and do occur in practice and to an extent that creates a serious error-tolerance design problem that must be taken into account. Erroneous bonds are weak bonds and are therefore more easily broken down, depending on environmental conditions. This fact can be exploited in devising error-correction techniques, as shown in the remainder of this chapter. Such techniques will circumvent a fundamental limitation to handling errors in molecular computation: the physical sizes of the computational structures make it almost impossible for humans or machines to intervene and control the errors.

Various methods have been used to deal with error-prone computers, including notably coding and redundancy techniques for detecting, preventing, or correcting errors and checkpointing techniques for graceful recovery from errors. This last method, which this section adapts to rectangle self assembly, determines regular checkpoints,
or milestones, at which the state of a computation, or the system as a whole, is stored in a reliable memory. When an error occurs the state of the computation can be restored from, i.e., by rolling back to, the state saved at the most recent checkpoint. See Chapter I.1 for a broader introduction and literature review of error correction in molecular computing.

4.2 Checkpointing and Roll-back via Pulsing

To apply checkpoints and roll-back recovery to error-tolerant self-assembly, the process is set up to repeatedly grow the crystal during growth stages of fixed duration \( \tau \). At each checkpoint, i.e., at the end of each growth stage, a positive pulse in temperature is made. This pulsing process is engineered so that it breaks down the weaker hydrogen bonds between the sticky ends of the erroneously bonded molecules (to be called simply error tiles in the tile model), but has no effect on the corresponding bonds of the correctly assembled molecules. Thus, the roll-back process of tile self-assembly can be viewed simply as the detachment of those corrupted parts of the crystal lying above and to the right of error tiles.¹ Figure 4.1 illustrates the roll-back process.

The temperature pulsing method applied to rectangle self assembly repeats the growth stages, temperature pulses, and roll-backs until the profile following a temperature-pulse passes through or beyond the desired location, say \((M, N)\), at the upper right corner of the rectangle. In addition to \(M\) and \(N\), the parameters of the process include the time \(\tau\) between checkpoints, and the probability \(q\) of an erroneous tile attachment which, as is evident from our notation, we take as independent of the locations of available sites and the properties of the checkpointing procedure. To focus the analy-

¹Note that saving a state is not an explicit operation in the current application of checkpointing and recovery; the saved state may be thought of as the subset of the current crystal under the most recent error-free profile.
sis, we assume that the rectangle is in fact an $N \times N$ square. Extending the results to general rectangles is routine, and discussed briefly at the end of this chapter. Another parameter to be considered later is the time, $p_s$, taken for the temperature pulse and the detachment of erroneous tiles. For the moment, we take this quantity to be 0.

The profiles at the beginnings of growth stages, i.e., just after temperature pulses, are called checkpoint profiles and are of particular interest in our analysis of checkpointing self assembly. Note that the available sites of a checkpoint profile are sites that were occupied by error tiles just before a temperature pulse. These error tiles are called seed error tiles and are responsible for corrupting the part of the crystal that grows up and to the right of the seed error tile and subsequently detaches. During a growth stage there will be many other error tiles in general, but clearly these do not have a corruptive influence beyond that of the seed error tiles already in place. Because of the Bernoulli law\(^2\) governing the creation of error tiles, the checkpoint profiles form a two-dimensional Markov process; but this process is significantly more complicated than self assembly without checkpointing. It is obvious that these processes cannot be mapped directly onto one-dimensional particle processes with local interactions (such as the TASEP). Hence, we are limited to asymptotic analysis and experimental studies in order to gain insights into checkpoint self assembly. The remainder of this chapter establishes an asymptotic relationship between the sequence of checkpoint profiles and the Hammersley process introduced in Chapter 1, and from this relationship determines the expected number of temperature pulses needed in the checkpoint self assembly of rectangles. The accuracy of these results will be tested experimentally at the end of this chapter. Further experiments are described which evaluate other aspects of checkpoint self-assembly performance not amenable to analysis.

\(^2\)With $q$ the error ("success") probability, we adopt a standard abbreviation and call the process of creating error tiles a $q$-Bernoulli process.
Let \( L \equiv L(N, q, \tau) \) denote the number of growth stages needed in the checkpoint self assembly of an \( N \times N \) square. A sequence of tile locations\(^3\) \((x_1, y_1) \cdots (x_k, y_k)\) is said to be increasing if it is coordinate-wise increasing, i.e., \( x_i < x_{i+1}, \ y_i < y_{i+1} \) for every \( i, 1 \leq i < k \). As before, the checkpoint profiles are monotonically non-increasing. Moreover, the sequence of checkpoint profiles is monotone in the obvious sense guaranteed by the roll-back process: the heights of the profiles above \((i,0)\) are non-decreasing for all \( i \geq 1 \); similarly, the widths of the profiles to the right of \((0,j)\) are nondecreasing for all \( j \geq 0 \). In other words, the \((i + 1) - st\) checkpoint profile is at or beyond the \( i-th \) at every point. The next observation is key:

The number \( L \) of growth stages in the checkpointing self assembly of an \( N \times N \) square is just the length of a maximum-length increasing sequence of seed error tile locations within the square.

Figure 4.2 shows an example. The longest increasing sequence in this example is unique, but in general there will be more than one such sequence. The observation follows from the monotonicity of the checkpoint profiles: with reference to Figure 4.2, for each corner in the \((i + 1)\)-st profile there must be at least one corner in the \( i-th \) profile below and to its left. This guarantees a sequence of corners corresponding to a longest increasing subsequence spanning all checkpoint intervals.

The reader may have noted the resemblance between the profiles of Figure 4.2 and the trajectories of the graphical representation of the Hammersley process (cf. Chapter 1). The generators are quite different, but if we can show that the pattern of seed tile locations is approximately Poisson, then we can apply the Aldous-Diaconis result for maximum-length increasing sequences in the graphical representation of the Hammersley process. This can be done in an asymptotic regime of small error rates

\(^3\)Recall that these are given by the coordinates of the upper right corners of the tile sites.
CHAPTER 4. ERROR-CORRECTION

$q$ as shown in the next section.

First, however, we need to verify that the $q$-Bernoulli law governing error-tile attachments implies that the pattern of seed error-tile locations also comprise such a $q$-Bernoulli pattern. A short proof of this assertion invokes as follows the Principle of Deferred Decisions. In addition to the seed-error tiles that define the $i$-th checkpoint profile, there will in general be many other tiles, both good and erroneous, that attach before the $(i-1)$-st growth stage is complete (at which point the corrupt portion of the crystal detaches, leaving the $i$-th checkpoint profile as the initial state of the $i$-th growth stage). The growth to proceed from the $i$-th checkpoint profile which establishes the seed error tiles defining the $(i+1)$-st checkpoint profile will have been "deferred" by much of the process that established the $i$-th checkpoint profile. By the Principle of Deferred Decisions, the probability laws (the memoryless exponential tile attachment times and $q$-Bernoulli error-tile selection) driving the growth of the $i$-th stage will be unchanged by the conditioning events deferring this growth. To apply this principle, it is enough to observe that the information supplied by the knowledge of the deferring growth (attachment locations of both good and erroneous tiles) does not change the probability law determining future growth.

4.3 Small $q$ Asymptotics

To calculate the times required by our error-correcting self-assembly process, we apply classical estimates for the expected length, $L$, of a maximum-length increasing subsequence of seed-error tile locations, as discussed in Chapter 1. To do this, we need to scale the $q$-Bernoulli seed-error process so as to approximate a unit-intensity Poisson seed-error process (cf. e.g., [28]). This immediately implies recourse to small-$q$ asymptotics. Precisely, scale down the problem by reducing the tile sites (lattice

\footnote{This principle originated in the analysis-of-algorithms field (cf., e.g., [33])}
squares) by a factor of $\sqrt{q}$, and scale the growth stage duration similarly to $\tau_\star := \tau \sqrt{q}$. We now have an $N \sqrt{q} \times N \sqrt{q}$ square of $\sqrt{q} \times \sqrt{q}$ tile sites still with $N^2 q$ seed-error tiles on average, but a unit intensity of seed-errors. For small $q$, large $n := N^2 q$, and large $\tau_\star$, the original $q$-Bernoulli seed-error tile patterns are approximated by unit-intensity Poisson patterns of $n$ points. This in turn gives us the basis for applying the estimate from Chapter 1:

$$L \approx 2\sqrt{n} = 2N \sqrt{q}$$

Precisely,

$$L_{q, N, \tau} \sim 2N \sqrt{q}$$

as $q \to 0$ together with $N^2 q, \tau^2 q \to \infty$. 

Figure 4.1: The profile of the lattice before a pulse (dashed line) and after a pulse (solid line).
Thus, multiplying by the duration of a growth stage, we get an estimate of

\[ C_{N,N}(q) \sim 2\tau N \sqrt{q} \]

for the total self-assembly time of an \( N \times N \) square in a system with error rate \( q \). We note that in the asymptotic regime the number of growth stages does not depend on \( \tau \). This could have been anticipated, since the size of \( \tau \) beyond that which is needed to supply a sizable fraction of the seed error tiles has little effect. This will be quantified by experiments described below, which are designed to yield good values of \( \tau \).

4.4 Experiments

We begin with simulations that give total tile-attachment times, \( T_A \), to complete the checkpointing self assembly of a 500 × 500 square. The objective will be to study
behavior as it varies with the parameters $\tau$ and $q$.

First, however, we describe briefly algorithmic issues. For simplicity, consider an algorithm that includes the entire $500 \times 500$ array of cells (lattice squares) in the state, with all cells initially marked as empty except for the seed cell which is marked as occupied. At the start of each growth stage, a timer is started with the initial value $\tau$. While the timer is running, an algorithm similar to the randomized self-assembly algorithm described in Chapter 1 is executed with the following modification: For each tile attachment, a biased coin is tossed so that the cell is marked as an error cell with probability $q$; otherwise, the cell is marked simply as occupied. When the timer runs out, the array is scanned in left-to-right, bottom-up order for cells marked as errors. When one is encountered, it and all subsequent cells in the same row are marked as empty. When all rows have been processed in this way, the cell at position $(500,500)$ is checked; if it is marked as occupied the algorithm halts, but otherwise, a new checkpointing stage begins with the new initial state.

Figure 4.3 shows the effects of $\tau$ and $q$ on the performance of our error-tolerant self-assembly method. The data show that as the control parameter $\tau$ is increased for fixed $q$, the number of pulses needed flattens out abruptly beyond the knee of the curves roughly in the region $5 \leq \tau \leq 10$. It is in this region where $\tau$ should be chosen for small self-assembly times. We return to this issue a little later.

Next, we account for pulsing overhead which is directly proportional to the number of pulses required to self-assemble the $500 \times 500$ square. Figure 4.3(b) shows the number of pulses required for various values of $\tau$ and $q$. Small values of $\tau$ lead to large values of $L$, so when the pulsing-plus-detachment time $p_s$ is non-negligible, large pulsing overhead times result. Since the total time required to complete the crystal, $T$, is $T_p + T_A$, we can see from the behavior of $T_A$ in Figure 4.3(a) and that of $T_p$ in
Figure 4.3: The performance of checkpointing self assembly of a $500 \times 500$ square as a function of $\tau$ and $q$. 
Figure 4.3(b) that $T$ will have an optimal point for a given $p_s$.

The total time required to complete the $500 \times 500$ lattice is the sum of $T_A$ and $T_p = p_s L_p$. Figure 4.4(a) shows the total-time surface plot for the different values of $\tau$ and $q$. For simplicity, we assume that the time required for each pulse, $p_s$, is linearly proportional to the growth time, $\tau$. As an arbitrary example, we take $p_s = 0.2 \tau + 2$ to show how $p_s$ can affect the total time, $T$. According to the figure, for a given value of $q$, there is an optimal $\tau$ that minimizes the total time to complete the DNA self-assembly process. Figure 4.4(b) shows the total time with different values of $\tau$ when the probability $q$ of a tile being in error is 0.05. The figure shows that we can obtain the smallest lattice tiling times when $\tau$ is approximately 9 time units.

For all sufficiently small $q$ used in our simulations, the total number of pulses required to complete the self-assembly of a $500 \times 500$ square was in close agreement with our estimate in Chapter 4:

$$L_p \sim 2N\sqrt{q}$$  \hspace{1cm} (4.1)

Figure 4.5 shows results from experiments verifying the above relationship. In Figure 4.5(a), the assumptions are self-assembly of a $500 \times 500$ square and a growth duration $\tau = 500$. In Figure 4.5(b), the error rate $q$ is 0.005 and the growth time $\tau$ is 800. According to the results, $L_p$ is directly proportional to $\sqrt{q}$ and $N$, which is in line with our asymptotic estimate.

However, as expected, as $q$ becomes large, the number of pulses required to complete the $N \times N$ square no longer follows (4.1). For example, consider the value of $L_p$ for $q$ as large as 0.1 with $N = 500$, $\tau = 500$. Figure 4.7 and 4.3(b) illustrate the deviation of $L_p$ from the small-$q$ estimate. Based on the experimental results in Figures 4.7 and 4.6, the estimate 4.1 is a useful approximation for $q \leq 0.01$. In Figure
Figure 4.4: Figure 4.4(a) shows the total-time performance of the self-assembly of $500 \times 500$ squares with pulsing for different values of $\tau$ and $q$ where $p_s = 0.2 \tau + 0.2$. Figure 4.4(b) shows the total-time performance of the same self-assembly system for different values of $\tau$ and $q = 0.05$. 
Figure 4.5: Simulations show that $L_p$ is directly proportional to $\sqrt{q}$ and $N$. This agrees with our analysis $L_p \sim 2N\sqrt{q}$.

4.6, the error is represented as a percentage of the width $N$ of the structure.

Even though (4.1) does not show an explicit dependence on the value of $\tau$, recall that this independence exists only when $\tau$ is sufficiently large. If the value of $\tau$ is too
CHAPTER 4. ERROR-CORRECTION

Figure 4.6: For the number of pulses required to complete the crystal of size $500 \times 500$, simulations show that $q$ should be lower than 0.01 for our asymptotic estimate to be accurate.

Figure 4.7: Simulations show that $L_p$ is directly proportional to $\sqrt{q}$ only for small $q$.

small, the seed error-tile distribution departs from the Poisson, particularly near the boundaries of the structure, and the number of pulses does indeed depend on $\tau$. Our
present interest is in finding what values of $\tau^*$ are sufficiently large. Figure 4.8 illustrates the error percentage relative to $N$ for the self-assembly of a $500 \times 500$ crystal as $\tau$ varies. The data suggest that $\tau$ should be greater than 50; beyond that point, the number of pulses required to complete the crystal does not depend significantly on $\tau$.

![Graph](image)

Figure 4.8: For the simulation of a $500 \times 500$ crystal, when $\tau$ is greater than 50, the number of pulses does not depend significantly on the value of $\tau$. 
Appendix

The technical details for the experiments performed throughout this thesis are summarized here. The algorithms for the simulations of self-assembly models are applied in Part II (Chapter II.2 for irreversible self-assembly, and Chapter II.5 for reversible self-assembly). The algorithms for the tile self-assembly simulations are applied in Part III (Chapter III.2 for the growth process of tile self-assembly, and Chapter III.4 for the pulsing error-correction method). It bears repeating that, even though self-assembly systems in the real world consist of a vast numbers of elements, our simulations have been shown to produce results that are strikingly close using much smaller numbers of elements (many orders of magnitude smaller). In what follows, we briefly discuss technical issues that arise during our experiments and outline methods to speed up experiments. For the source codes of our simulations, the interested reader can submit a request directly to the author.

Random Number Generators

Our simulations are only as good as the (pseudo) random number generators we have used. We used Java for most of the simulations performed in this thesis except the simulation of tile self-assembly systems with pulsing error-correction, in which we used C++. Our decision to use C++ or Java was just a matter of convenience. For every Java implementation, we used the standard random number generator Math.random() built into the Java standard package. The Random class in
the `java.util.Random` package included in version 1.2 and higher of Java is more organized and more suitable for object-oriented programming. However, the implementations of our simulations did not require object-oriented programming. Thus, we decided to stay with the traditional `Math.random()` method which offers the same quality of random numbers as the `Random` class in the `java.util.Random` package. The `Math.random()` uses 48-bit seeds, which are modified using a linear congruential formula (see e.g., [35]).

There are random number generators that are better than the `Math.random()` method such as the `SecureRandom` class in the `java.util.Random` package and `Mersenne Twister` random number generator. Both generators are known to have a very long period and to be suitable for cryptography. However, we decided to use the `Math.random()` because of its ready availability and its excellent performance record. Also, speed and complexity are superior to the alternatives.

For our C++ implementation, we used the ANSI C/C++ `rand()` function to model uniformly distributed random numbers. The basis for our decision to use the ANSI C/C++ `rand()` was the same as for our decision to use the standard `Math.random()` in Java. Unlike the Java `Math.random()` method, the ANSI C/C++ `rand()` function does not generate its own seed; for this, we used `srand(time(NULL))`, which uses the current time in seconds to initialize the sequence.

**Convergence Rates of Self-Assembly**

Our experiments in Part II focused on incremental self-assembly where systems start with \( \eta(0) \) monomers. According to our self-assembly algorithm discussed in Chapter II.2, the colliding polymers are chosen independently from the distribution \( \{ \eta_1(t)/\eta(t), \ldots, \eta_N(t)/\eta(t) \} \). The two polymers, say \( i \)- and \( j \)-polymers bond only if at least one is a
monomer, and then only with probability \( p_j \) or \( p_k \), depending on which is a monomer. The process continues until there is no monomer left in the system. A straightforward implementation of the simulation of self-assembly models discussed in Chapter II.2 eventually approaches an absorbing state extremely slowly; when the initial number \( \eta(0) \) of monomers is large, then when in a state where relatively few remain, a great many collision samples are made on average before two reacting polymers (one or both must be a monomer) are encountered; indeed, if \( \eta(0) \) is large enough, it becomes impractical to wait for entry into an absorbing state.

In order to avoid this convergence problem when the number of monomers becomes small, we can compute the (geometric) distribution of the number of samples between successive reactions, then use a sample of this distribution to skip over the large number of collision samples where no reaction takes place. One can easily estimate a threshold on the number of remaining monomers, such that, when this number drops to the threshold, the simulation switches over to the "skipping mode" for the remainder of the simulation.

That said, we remark that, having set \( \eta(0) \) to be no larger than 50000 in all of our experiments, all of our simulations reached an absorbing state with no monomers and did so without undue waiting times.
Bibliography


