Active Self-Assembly of Simple Units Using an Insertion Primitive

Nadine Dabby * Ho-Lin Chen †

Abstract

While computer science has given us a framework for determining the complexity and difficulty of solving computational problems, we do not yet have a theoretical framework for knowing what actions, behaviors, and life-like qualities can emerge from a given set of simple modular units. There has been much interest in developing models for programming active self-assembly processes in both the reconfigurable robotics community and the nanotechnology community. With respect to materials science and nanotechnology, the models proposed to date are either not yet implementable with our current understanding of synthetic chemistry or those that are implementable are limited to a set of features that do not capture the power of active components. Prior implementable models of molecular assembly only considered the passive behaviors of attaching and detaching from a complex.

Inspired by the algorithmic tile assembly model [Winfree, 1996] and the graph grammar assembly model [Klavins et al., 2004], we describe a formal model for studying the complexity of self-assembled structures with active molecular components. In particular, we add an insertion primitive and we show a direct mapping of our model to a molecular implementation using DNA. We show that the expressive power of this language is stronger than regular languages, but at most as strong as context free grammars. Here, we explore the trade-off between the complexity of the system (in terms of the number of unit types), and the behavior of the system and speed of its assembly. We find that we can grow a line of any given length n in expected time $O(log^3 n)$ using $O(log^2 n)$ monomers. If we grow a line with k insertion rules, either the expected final length is infinite or the expected length at time t is at most $(2t+2)^{k^2}$, which is polynomial in t.

1 Introduction

Molecular programming, nanotechnology and synthetic biology raise the prospect of bottom-up fabrication, the manufacture of complex devices from simple components that assemble themselves. Biology sets the bar high: fabricating systems of enormous scale, defined at atomic-scale resolution, that grow quickly with small programs relative to object size and algorithmic complexity [21]. A human's genome consists of approximately 3 billion base pairs [45]; this implies that our cells are running a program that utilizes less than 1 Gigabyte of information. Contrast this program-size efficiency, with the computer on which we write this report: it has 320 Gigabytes of memory, and yet it is not capable of doing many things that biology can do (e.g. it cannot grow exponentially fast like the embryos shown in Fig. 1, it cannot grow in mass and develop simultaneously, and it is not robust to damage). Other examples from biology prove to be even more phenomenologically interesting: a newt is able to regenerate its tail, a flatworm is capable of regenerating its head, and a starfish can regenerate its entire body from a severed leg [2]. Biology offers many examples of phenomena that we are as of yet unable to reproduce in computational software or hardware, but that perhaps show us what is possible. Inspired by these feats of biological efficiency, robustness and phenomena, we define a formal implementable model for active self-assembly.

Many attempts have been made to emulate biology's success across materials and disciplines. While biologists have had success reconstructing self-organized cellular systems in vitro [25], chemists have utilized self-assembly to construct films and monolayers as well as more complicated architectures constructed from nanotubes and nanoparticles [17, 49]. These new self-assembled materials have in turn been used to construct nano-scale electronics [26] and biomaterials [42].

Nanotechnologists have built many examples of self-assembling 2 and 3 dimensional devices using passive subunits. The nano-components of a cell are much more "active" than passive: they sense and process environmental cues; they assemble and disassemble; upon interaction, their configurations often change, determining their future interactions; they can both diffuse and actively move. Recently, self-assembly systems using active molecular components have also been demonstrated in various synthetic systems [5, 18, 19, 22]. Particularly notable are the rich dynamical systems constructed out of synthetic nucleic acids, whose four-base code gives rise to a means of programming specific molecular interactions. DNA has been used to build autonomous walkers [4, 14, 28, 31, 33, 34, 43, 55, 56], logic and catalytic

*Department of Computation & Neural Systems, California Institute of Technology. Email:ndabby@caltech.edu. Research supported by an NSF Graduate Research Fellowship, NSF grant CCF-0829805, and the Molecular Programming Project under NSF grant 0832824.

†(corresponding author) Department of Electrical Engineering, National Taiwan University. Email:holinc@cc.ee.ntu.edu.tw. Research supported by NSC grant number 101-2218-E-002-007-, National Taiwan University Electrical Engineering Department, NSF grant CCF-0829805, the Molecular Programming Project under NSF grant CCF-0832824, and the Caltech Center for the Mathematics of Information (CMI).

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

circuits [39, 50, 55, 58], and triggered assembly of linear [10, 44] and dendritic structures [55].

Now that our once passive subunits can actively sense, walk and otherwise actively interact, how do these new "rules" change the prospects for what we can build from the bottom-up? This notion of actively assembling molecules is already an experimental reality, but as of yet there is no satisfying theory to guide future work. In this paper we attempt to formulate a framework for integrating these new "active" mechanisms in nanotechnology to build "programs" that can grow into a desired shape quickly and with relatively small program size to a final structure.

After reviewing prior self-assembly models and constructions across disciplines in the remainder of this section, we describe our model for active self-assembly utilizing a simple insertion primitive in Section 2. In Section 3, we show that the expressive power of this language is stronger than that of regular languages but at most as strong as the expressive power of context free grammars. In Section 4 we show a construction for building a line in logarithmic time using a logarithmic number of monomers and map it to a molecular system. To our knowledge this is a novel assembly system that has not been synthetically constructed before. Our three main results are: (1) Given any insertion system, the expressive power of this language is stronger than regular languages, but at most as strong as context free grammars. (2) We find that we can grow a line of any given length n in expected time $O(log^3 n)$ using $O(log^2 n)$ monomers. (3) If we grow a line with k insertion rules, either the expected final length is infinite or the expected length at time t is at most $(2t+2)^{k^2}$, which is polynomial in t.

1.1 Review of Self-Assembly Models The Tile Assembly Model integrates the algorithmic association of units with a defined geometry: the exposed edges of a growing crystal encode the state information of the system, and this information is modified as a new tile attaches itself to the crystal [51]. This model formally couples computation with shape construction, and the shape can be viewed as the output of the tile assembly "program". Tiles are capable of universal computation [51]. The system can grow an arbitrary shape (independent of scale) using a tile program whose complexity, defined as the number of distinct tile species in the program, is bounded from both above and below by the shape's descriptional (Kolmogorov) complexity [41]. The time required to build an $n \times n$ shape through passive self-assembly is $O(n)$ [1]. This bound can be improved to $O(n^{4/5} log(n))$ with massive parallelism [8]. In this model, scale plays the same role in the self-assembly process as time plays in computability. While the Tile Assembly Model is elegant in its simplicity and ability to capture experimental reality, it is limited in its speed, its ability to be scaled up and its focus on passively assembling units.

Drawing on cellular automata and Chomsky grammars, L-systems were developed as a theoretical framework for studying development in multicellular organisms and were one of the first models used to simulate growth and development in plants [24]. Although they bear a resemblance to cellular automata, they differ in that arrays can grow and shrink (introducing the notions of insertion, a new cell is generated by division of a prior cell, and deletion, the elimination of a cell). L-systems differ from grammars in that they require parallel rewriting of all symbols and do not distinguish between terminal and non-terminal symbols [24]. While these models are well-developed for one-dimensional systems, they have also been studied in 2 [40] and 3 dimensions [35]. While L-systems have aided in the modeling of plant growth and biology, the formal work does not address theoretical questions related to the complexity of pattern formation such as how quickly a system can generate a specific pattern.

A number of geometric models and numerous algorithms have been described for self-assembling and reconfigurable modular robotic systems [3,7,9,13,15,16,20,30, 32, 36, 37, 46–48, 53, 54]. Existing formal models haven't fully captured the efficiency of active self-assembly: to assemble a prescribed shape, most of the models require a linear (to the size of the shape) number of distinct states.

One of the central questions that this work addresses is how to program global tasks through local interactions. Our approach is inspired by Klavins' work on modeling robotic self-assembly [23] using conformational switching [38] and graph grammars [11]. In Klavins' work, the state of a physical system is represented as an abstract graph, where an assembly unit is represented as a symbolic vertex labeled with its current state, and the attachment of two units is represented by an edge connecting the two corresponding vertices in the graph. Assembly proceeds following graph rewriting rules, which update the system state by updating the vertex labels and edges of a subgraph under suitable conditions. This approach nicely captures the local, asynchronous, cooperative and conditional state change logic, which is intrinsic to assembly systems with active components, and it captures disassembly in addition to assembly. However, unlike the Tile Assembly Model, the graph grammar model represents the assembly system as an abstract graph, and leaves out geometry, which is a crucial property for the assembly of physical systems.

In our prior work on active self-assembly, we con-

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

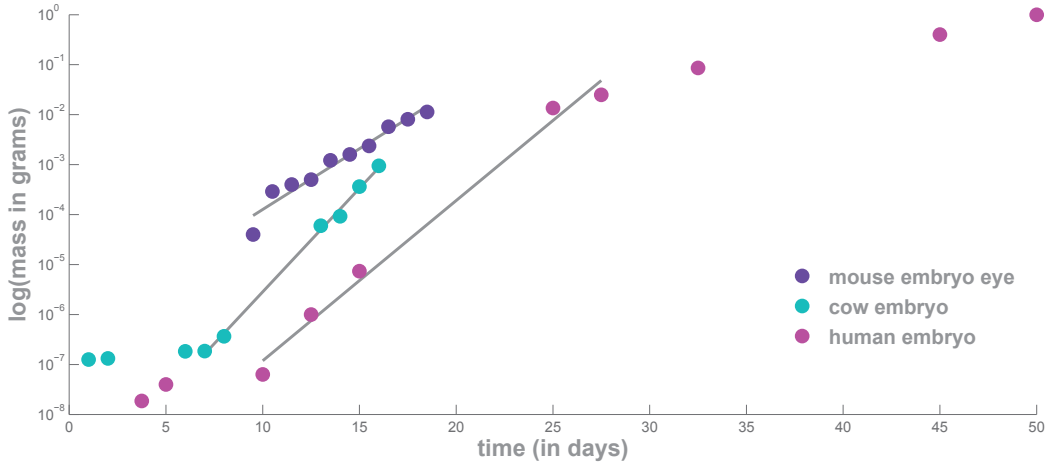

Figure 1: This plot is compiled from embryonic mouse [12], cow [29], and human [27] data. The gray lines fit the periods of exponential growth in each species. Note that beginning points do not reside on these lines, because the growth rate initially increases proportionally to mass. The period of exponential growth slows down as the amount of mass necessary to sustain this type of growth becomes constrained by volume and access.

structed the "nubot" model by adding a geometric component to the graph grammar model [52]. The nubot model builds on the concept of graph grammars, by defining rule sets over two dimensional monomers, represented as disks of unit diameter centered on a point in a hexagonal grid. Two monomers can react with each other (according to a rule) to change state, make and break bonds, change relative position, appear and disappear. With this model, a line of length n can be constructed with $O(\log n)$ states, in $O(\log n)$ time. A computable shape of size $n \times n$ pixels can be constructed in time polylogarithmic in n. This is exponentially faster than systems composed entirely of passive components. While the nubot model is not chemically implementable, it highlights the fundamental efficiency advantage of active self-assembly over passive self-assembly. We seek to preserve the complex behaviors that the abstract nubot model can generate, but in a formulation that is simple enough to implement experimentally.

2 Model

2.1 Formal Model Description In our model, each construction begins with an initiator, and grows via the insertion of simple units that we call monomers. We assume that each type of monomer in the system is present in infinite amounts. Monomers can be inserted into the middle of the structure and increase the length of the structure (an abstraction of the model is shown in Fig. 2). Figure 3 shows an example system that grows exponentially fast. The detailed description of initiators, monomers, and the insertion rules follows:

1. We have two finite sets of symbols $\Gamma = \{a_1, a_2, a_3, a_4, \dots\}$ and $\Gamma^* = \{a_1^*, a_2^*, a_3^*, a_4^*, \dots\}$. Each pair a_i and a_i^* are called *complementary* to each other.

2. There are k monomers, each is described by a quadruple of symbols (a, b, c, d) and either a plus sign or a minus sign. (For example, $(a_4, a_7, a_6^*, a_1)+$ or $(a_5, a_7, a_2^*, a_3^*)-$.) Each monomer has a concentration c. We assume that the total concentration is at most 1.

3. The initial state can be described by two pairs of symbols $(a, b), (c, d)$. Either a and d are complementary to each other or b and c are complementary to each other. Each of these pairs is considered a monomer.

4. An *insertion site* can only exist between two consecutive monomers: e.g., in the initial state (a, b) and (c, d) belong to two different monomers.

5. Only the following insertion rules are possible:

   (a) If there are two consecutive monomers connected in the structure such that the first one ends with the pair (e, a^*) and the second one starts with the pair (d^*, f), where e and f are complementary with each other, then any monomer of the form $(a, b, c, d)+$ can insert between those two groups, and add a

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

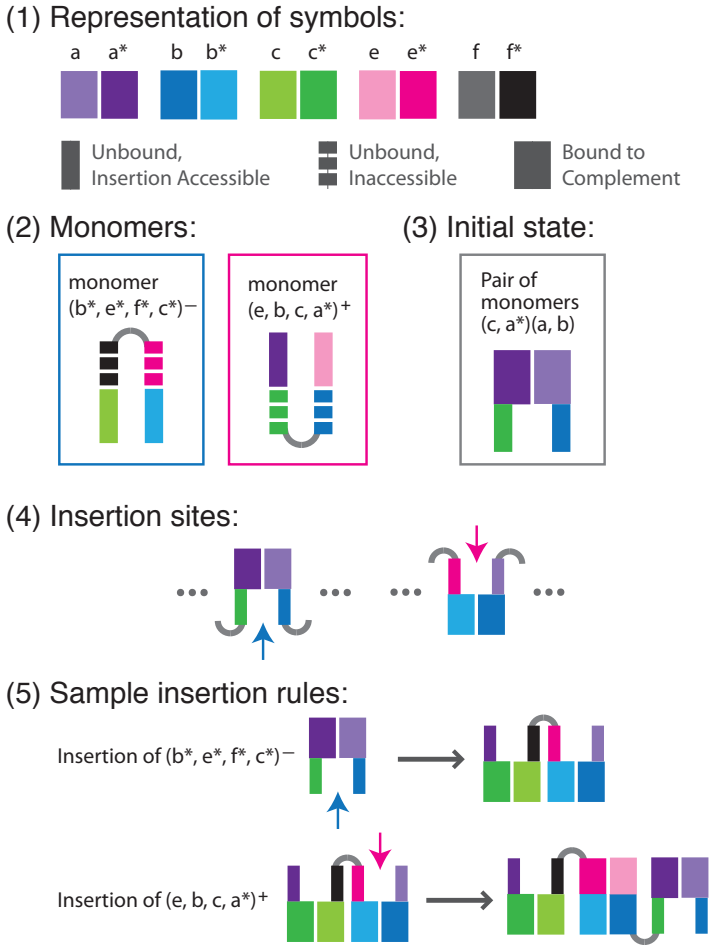

Figure 2: This figure depicts an abstraction of our model. (1) Each unique symbol is encoded by a color, and complementary symbols are represented by different shades of the same color. The symbols are represented as thin solid lines (Unbound, Insertion Accessible), thin dashed lines (Unbound, Inaccessible), and thick solid lines (Bound to Complement). (2) Two sample monomers are $(b^*, e^*, f^*, c^*)-$, and $(e, b, c, a^*)+$. (3) The initial state is described by the pair of doubles $(c, a^*), (a, b)$. (4) Insertion sites can only exist between two consecutive monomers connected in the structure; we use colored arrows to denote these sites. (5) Sample insertion rules show the insertion of monomer $(b^*, e^*, f^*, c^*)-$ into $(c, a^*), (a, b)$ to generate the polymer $(c, a^*), (f^*, c^*, b^*, e^*), (a, b)$, and the insertion of monomer $(e, b, c, a^*)+$ into $(c, a^*), (f^*, c^*, b^*, e^*), (a, b)$ to generate the polymer $(c, a^*), (f^*, c^*, b^*, e^*), (e, b, c, a^*), (a, b)$.

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

group of symbols (a, b, c, d) in the middle. $(e, a^*), (d^*, f)$ is called an *insertion site*.

(b) If there are two consecutive monomers connected in the structure such that such that the first one ends with (d^*, e) and the second one starts with (f, a^*), where e and f are complementary with each other, then any monomer of the form $(a, b, c, d)-$ can insert between these two groups and add a group of symbols (c, d, a, b) in the middle. $(d^*, e), (f, a^*)$ is called an *insertion site*.

6. If a particular insertion is applicable, it occurs at time x, where x is an exponential random variable with rate c, where c is the concentration of the monomer inserted.

7. A *polymer* is a sequence of tuples of symbols reachable from the initial state, where the first and last tuples are pairs of symbols and the middle tuples are monomers (as defined in rule 2). A *terminal polymer* is a polymer such that no monomers exist in the system that can be inserted at any of the insertion sites available on that polymer. The *length* of the polymer is defined as the number of monomers that it contains.

2.2 A Molecular Implementation Given any system described above, there is a direct implementation of monomers into a set of DNA molecules. By encoding the order of the nucleotides in a DNA sequence, we can control the interaction of DNA strands. Subsequences of these strands are called domains and it is their binding (hybridization) and unbinding (disassociation) from complementary domains that determines what a system can do. In DNA nanotechnology, dynamic systems of DNA molecules can be controlled by toeholds, the short sequences of DNA that are complementary to single stranded domains in a target molecule [57, 59]. Toeholds serve as the inputs to dynamic DNA systems and initiate branch migration processes, the random walk process of bond breaking and formation that results in the exchange of one strand in the duplex for another single strand with the same sequence.

Our DNA implementation is inspired by the Hybridization Chain Reaction system developed by Dirks and Pierce [10]. Their construction, which triggers the polymerization of DNA monomers, uses two single-stranded DNA hairpins that have the same 18 base-pair stem sequence and one toehold that is complementary to the other hairpin's loop sequence. These hairpins are caught in a kinetic trap that causes them to react with each other very slowly in the absence of an initiator. When the initiator is added to the solution of monomers, it binds to the toehold of the first hairpin and undergoes a strand displacement reaction that opens the hairpin. The newly exposed sticky end of the hairpin can then undergo a similar reaction with the second hairpin. The two hairpins will continue to polymerize until the concentration of monomers is exhausted. The system was modified to employ a four-way branch migration design to create an autonomous polymerization motor [44]. The metastable fuel hairpins from the Hybridization Chain Reaction system were modified to include an extra toehold, and the initiator strand was replaced by an initiator complex that is composed of an "anchor" strand and a "rickettsia" strand. Upon mixing, the first hairpin binds to the sticky ends of the anchor-rickettsia complex, initiating a four-way branch migration in which the rickettsia strand is passed from the anchor to the hairpin. The second hairpin then binds to the newly exposed sticky ends and the rickettsia strand is passed to the second hairpin. The rickettsia strand continues to be passed back and forth between newly added hairpins as the polymer grows in its wake.

Any system described in our model can be implemented by designing DNA hairpins and an initiator complex as follows:

If there is a monomer $(a, b, c, d)-$, we add a hairpin with domains (a, x, b, c, x^*, d), x is the long domain. If there is a monomer $(a, b, c, d)+$, we add a hairpin with domains (a, x^*, b, c, x, d). The initiator is (a, x^*, b) binding with (c, x, d). The insertion rules defined in the model correspond to all possible reactions that can happen in the corresponding molecular system.

Figure 3 shows an example of a molecular implementation of our system. In addition to the monomer $(a, b, c, d)+$ (or minus), we can also have a new type of monomer $(a, b)(c, d)+$ that we call a divider monomer. The reaction available for $(a, b)(c, d)+$ is exactly the same as that for $(a, b, c, d)+$, except that after $(a, b)(c, d)+$ inserts, the polymer will be cut between (a, b) and (c, d) and divided into two parts , as illustrated in Figure 4.

3 Expressive Power

In this section, we first ignore the rates of insertion and show that the expressive power of this insertion system is at most equivalent to context free languages. This result implies that we can simulate arbitrary tile systems that assemble a single line. From [6], we know that the insertion system can construct lines of arbitrary expected length with $O(1)$ monomers.

THEOREM 3.1. *Given any insertion system, the set of terminal polymers that can be generated forms a context free language.*

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

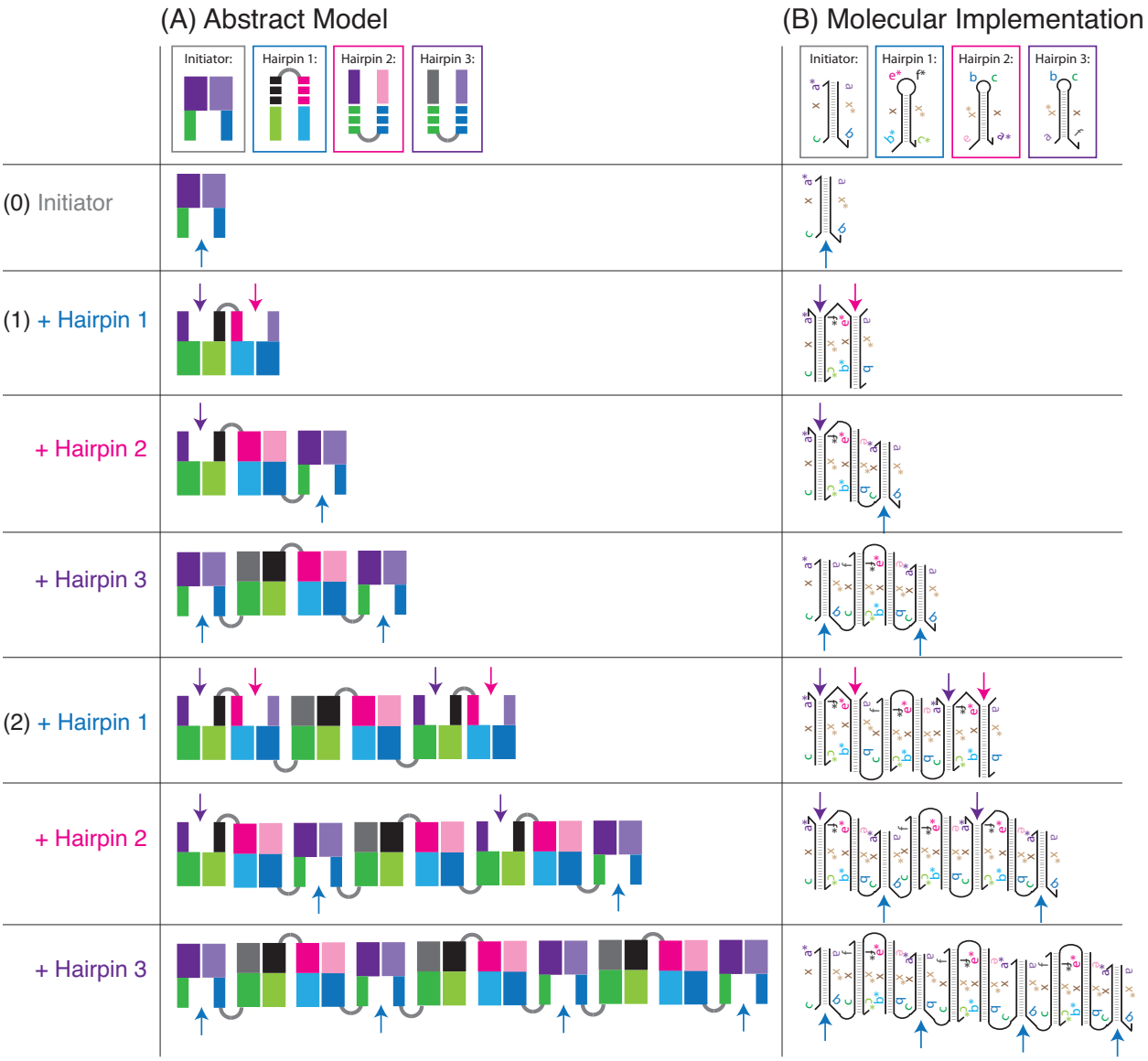

Figure 3: This figure depicts a system that implements insertional polymer growth in logarithmic time. The abstract representation of growth (A), is directly correlated to a molecular implementation (B). In this insertion system, the initiator is described as $(c, a^*), (a, b)$ and the three hairpins are $(b^*, e^*, f^*, c^*)-$, $(e, b, c, a^*)+$, and $(a, b, c, f)+$ After inserting hairpin 1, the polymer's description is $(c, a^*), (f^*, c^*, b^*, e^*), (a, b)$. After hairpins 2 and 3 are inserted, the polymer's description is $(c, a^*), (a, b, c, f), (f^*, c^*, b^*, e^*), (e, b, c, a^*), (a, b)$. The system will continue to grow to infinite length exponentially fast.

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

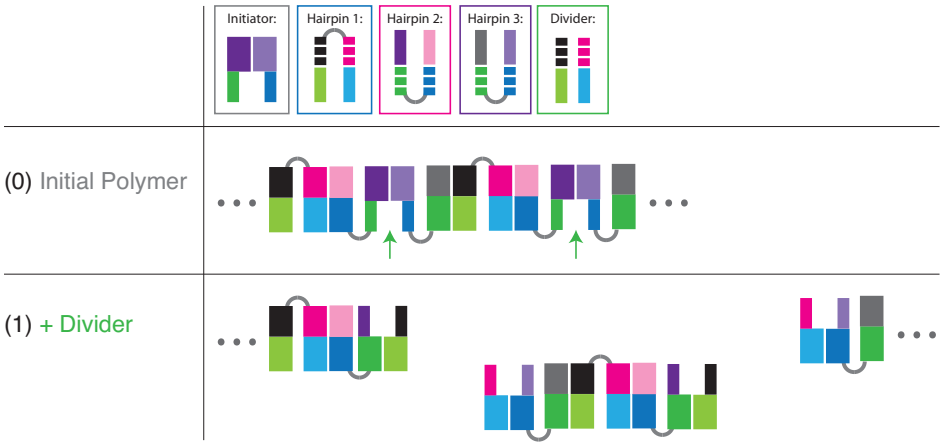

Figure 4: This figure depicts a system that implements division in a polymer. The reaction available for $(a,b)(c,d)+$ is exactly the same as that for $(a,b,c,d)+$, except that after $(a,b)(c,d)+$ inserts, the polymer will be cut between (a,b) and (c,d) and divided into two parts.

Proof. Given any insertion system with n symbols, the corresponding context free language has n^4 symbols, each of which corresponds to one insertion site. The starting symbol S corresponds to $(a,b),(c,d)$, which is the initiator of the polymer. Each monomer $(e,f,g,h)+$ corresponds to $2n$ different production rules in the context free language that starts with a symbol (insertion site) $(i,e^*),(h^*,j)$ and produces two symbols $(i,e^*),(e,f)$ and $(g,h),(h^*,j)$ for all possible choices of pairs of complementary symbols i,j in the insertion system.

THEOREM 3.2. *Given any regular language, there is an insertion system that generates terminal polymers corresponding to this language.*

Proof. Given any left regular grammar with nonterminal symbols $A_1,A_2,\ldots,A_n$, including the starting symbol A_1, and non-terminal symbols $\alpha_1,\alpha_2,\ldots,\alpha_m$, the following insertion system creates polymers that correspond to the given regular language:

1. $\Gamma = \{a_1,a_2,\ldots,a_n,b_1,b_2,\ldots,b_n,c_1,c_2,\ldots,c_m,d\}$.
2. The initiator is $(d^*,a_1),(d,d)$.
3. For each production rule $A_i \rightarrow \alpha_j A_k$, there are two corresponding monomers $(a_i^*,c_j,b_k,d^*)+$, $(d^*,a_k,d,b_k^*)-$.

In this system, there is always exactly one insertion site available at the end of the polymer. The insertion can only happen between two monomers (d,b_i^*,d^*,a_i) and (d,d). The insertion site between these two monomers corresponds to the nonterminal symbol A_i. At this point, two monomers $(a_i^*,c_j,b_k,d^*)+$, $(d^*,a_k,d,b_k^*)-$ may insert, generate some inactive sequence with j encoded in the middle and the end of the polymer becomes (d,b_i^*,d^*,a_k) and (d,d), corresponding to the nonterminal symbol A_k.

COROLLARY 3.1. *There is a family of insertion systems that can construct polymers of expected length n with $O(1)$ monomers.*

Proof. Since insertion systems are able to simulate all regular languages, they are able to simulate all tile systems that form a linear polymer of width 1. Therefore, the proof directly follows from [6], where the result was proven on 1-dimensional tile assembly systems.

4 The Growth Speed of Polymers

In this section, we will investigate the speed at which these polymers can be constructed. First, we show that arbitrarily long polymers can be constructed deterministically in expected polylogarithmic time using a polylogarithmic number of monomers.

LEMMA 4.1. *The following insertion system deterministically constructs a line of length $n = 2^{2k}+1$ in expected time $O(\log^3 n)$ and only uses $O(\log^2 n)$ monomers. Furthermore, the required time has an exponentially decaying tail probability.*

1. *The initiator is $(c,a_{2k}),(b_{2k}^*,c^*)$.*
2. *For every $i,j \in \{2,4,\ldots,2k\}, i \leq j$, there are two monomers $(a_i^*,b_{i-1}^*,a_{i-1},b_j)+$ and $(a_j^*,b_{i-1}^*,a_{i-1},b_i)+$. For every $i \in \{1,3,\ldots,2k-1\}$, there are two monomers $(b_i,a_{i-1},b_{i-1}^*,c^*)-$ and $(c,a_{i-1},b_{i-1}^*,a_i^*)-$.*

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

3. All monomers have equal concentration $\frac{1}{2k^2}$.

Proof. First, we show that the system deterministically constructs a line of length n. An insertion site of the form $(c, a_i), (b_j^*, c^*)$ is defined to have type $\min\{i, j\}$. Whenever a gap of type i is available, exactly one monomer of the form $(a_i^*, b_{i-1}^*, a_{i-1}, b_j)+$ and $(a_j^*, b_{i-1}^*, a_{i-1}, b_i)+$ will be able to attach. After the first monomer inserts, two monomers $(b_{i-1}, a_{i-2}, b_{i-2}^*, c^*)-$ and $(c, a_{i-2}, b_{i-2}^*, a_{i-1}^*)-$ will be able to insert on the first monomer's left and right. These three insertions create 4 insertion sites of type $i-2$. Therefore, starting with one insertion site of type k on the initiator, there will be $2^k - 1$ total insertions, resulting in a polymer that has n insertion sites of type 0. At that time, no further insertion is available and the system halts.

Second, the system halts as soon as all $\frac{n}{2}$ insertions of $(b_1, a_0, b_0^*, c^*)-$ and $(c, a_0, b_0^*, a_1^*)-$ happen. Each of these insertions only relies on k insertions to occur before them. Therefore, for any one of these insertions, the expected time T until the insertion occurs can be described as a sum of k independent exponential random variables of expected values $2k^2$. Using Chernoff bounds for exponential random variables, it follows that

$$\text{Prob}[T > 2k^2 \cdot k(1+\delta)] \ \leq \ (\frac{1+\delta}{e^\delta})^k.$$

Although the time required for these $\frac{n}{2}$ insertions are not independent of each other, we can still use a union bound to get the following bound for the total running time T_{fin} of the system:

$$\begin{aligned}\text{Prob}[T_{fin} > 2k^2 \cdot k(1+\delta)] \ &\leq \ \frac{n}{2}(\frac{1+\delta}{e^\delta})^k \\ &\leq \ \frac{n}{2}e^{-\frac{k\delta}{2}} \ < \ (\frac{e}{2})^{-\frac{k\delta}{2}}, \text{ for all } \delta > 4.\end{aligned}$$

Therefore, the expected time is $O(k^3) = O(\log^3 n)$ with an exponentially decaying tail probability.

THEOREM 4.1. *There exists an insertion system that deterministically constructs a line of length n in expected time $O(\log^3 n)$ and only uses $O(\log^2 n)$ monomers for every integer n. Furthermore, the required time has an exponentially decaying tail probability.*

Proof. Lemma 4.1 already showed that the theorem is true for all $n = 2^{2k} + 1$. Given an arbitrary n, we can write $n-1$ as the sum of $O(\log n)$ terms $2^{r_1} + 2^{r_2} + \cdots + 2^{r_m}$., where all r_is are even numbers. We can first construct m distinct monomers that must insert one by one at the beginning, creating m insertion sites identical to the initiator for a polymer of length $2^{r_i} + 1$. Afterwards, the system described in Lemma 4.1 can make a line of length n. Since m is only $O(\log n)$, this construction works in the required $O(\log^3 n)$ time and $O(\log^2 n)$ monomers.

In the rest of this section, the major goal is to show that for an insertion system with k different molecular species, either the expected final length is infinite, or the expected length grows polynomially with time.

THEOREM 4.2. *Consider a context free language L with m symbols (including terminal and nonterminal symbols) in reduced Chomsky normal form. When a production rule is applicable, the time until it is applied is a random variable of rate k. If, for any given symbol A, the rate of all production rules having A on the left side sum up to at most 1, then either the expected final length is infinite, or the expected number of symbols at time t is upper bounded by $(2t+2)^m$.*

Proof. Assuming the expected final length is finite, we will prove inductively on m that the expected number of symbols at time t is upper bounded by $(2t+2)^m$

The general idea is that starting with S, we can't produce S too fast, otherwise the expected length will become infinite. Furthermore, since L is a context free language, if all we want to know is the length of the string, we only need to keep track of how many copies of each symbol is currently in the string. If each time we generate S we isolate that symbol into a new sub-system, then each sub-system is essentially a system with $m-1$ different symbols and the growth speed will be bounded by the induction hypothesis.

The theorem is true when $m = 2$. Since there are only two symbols S, α, starting from S, if the rate of the production rule $S \to SS$ is higher than the rate at which $S \to \alpha$, then the expected length is infinite. Otherwise the expected length is linear in t, since the expected number of symbols S in a string is at most 1 at any given time.

Assume that the theorem is true for $m = k-1$. For $m = k$, we subdivide the sets of symbols into many subsets in the following way: initially, there is only one subset that contains S; whenever one copy of S gets produced in any subset, we move that symbol S into a new subset; when other types of symbols are produced, they stay in the same subset.

First, we show that the expected number of symbols in each subset is quite small at time t. We start by considering the subset T_1 that the initial symbol S belongs to. After applying the first production rule, the subset T_1 has at most 2 symbols and will never contain another copy of S again. Therefore, after that first production rule, only $k-1$ different symbols can appear in that subset. By the induction hypothesis, either the expected number of symbols in T_1 goes to infinity, or the expected number of symbols is upper bounded by $2 \cdot (2t+2)^{k-1}$. The exact same argument can be applied to all other subsets.

Second, we will show that the expected number of subsets at time t is at most t. Notice that the number of

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

subsets is equal to the number of symbols S that have been generated in the process. For the expected final length to be finite, the expected number of symbols S in the system is at most 1 at any given time. (Otherwise the number of symbols S is expected to grow exponentially, a contradiction.) Furthermore, since the total rate of all rules with S on the left side is 1, the expected rate at which S is removed by applying production rules is also at most 1 at any time. Therefore, at any time t, the expected number of symbols S that have been removed by a production rule is at most t. Combining the above arguments, the expected number of symbols S that have ever appeared in the system before time t is at most $t + 1$.

According to our definition, at time t, the expected number of subsets is equal to the expected total number of symbols S that have ever appeared in the system, which is at most $t + 1$. Also, the expected number of symbols in each subset is at most $2 \cdot (2t + 2)^{k-1}$. Using linearity of expectation, we know that the expected number of total symbols at time t is at most $(2t + 2)^k$.

COROLLARY 4.1. *Given any insertion system with k molecular species and total concentration 1, either the expected final length is infinite, or the expected length at time t is upper bounded by $(2t + 2)^{k^2}$.*

Proof. There are at most k^2 different insertion sites in a system with k species. From Theorem 3.1, we know that the insertion system can be described by a context free grammar in reduced Chomsky normal form with at most k^2 symbols. Therefore, the proof follows from Theorem 4.2.

5 Conclusions

We have defined a formal implementable model for active self assembly utilizing an insertion primitive. We build on the concept of applying biological algorithms to the development of novel techniques in computer science to provide a method by which we can program arbitrary insertion systems whether they be reconfigurable robots, molecules or scripts of symbols. The work here is particularly relevant for the application of computer science to synthetic biology, chemistry and material science. We show a construction for building a line in polylogarithmic time using a polylogarithmic number of monomers and map it to a molecular system. To our knowledge this is a novel assembly system that has never been synthetically constructed before. We also show that with a number of monomer types the system will either grow to infinity or the expected length of the polymer grows polynomially with time. There are many interesting open questions remaining: What other behaviors can be generated by such a simple model? Are there other directly implementable simple primitives that we can add to this model to generate such behaviors? In this paper we explored the expressive power of this language, and proved that the language is stronger than regular languages, but at most as strong as context free grammars. It remains to be shown whether this system is equivalent to context free grammars, in which case the language will prove to be even more powerful than we suggest here.

References

[1] L. ADLEMAN, Q. CHENG, A. GOEL, AND M. HUANG, *Running time and program size for self-assembled squares*, in Proceedings of the thirty-third annual ACM Symposium on Theory of Computing, ACM, 2001, pp. 740–748.

[2] A. ALVARADO AND P. TSONIS, *Bridging the regeneration gap: genetic insights from diverse animal models*, Nature Reviews Genetics, 7 (2006), pp. 873–884.

[3] D. ARBUCKLE AND A. REQUICHA, *Active self-assembly*, IEEE International Conference on Robotics and Automation, 1 (2004), pp. 896–901.

[4] J. BATH, S. GREEN, AND A. TURBERFIELD, *A free-running DNA motor powered by a nicking enzyme*, Angewandte Chemie International Edition, 44 (2005), pp. 4358–4361.

[5] J. BATH AND A. TURBERFIELD, *DNA nanomachines*, Nature Nanotechnology, 2 (2007), pp. 275–284.

[6] F. BECKER, I. RAPAPORT, AND E. RÉMILA, *Self-assemblying classes of shapes with a minimum number of tiles, and in optimal time*, in FSTTCS: Foundations of Software Technology and Theoretical Computer Science, Springer, 2006, pp. 45–56.

[7] Z. BUTLER, K. KOTAY, D. RUS, AND K. TOMITA, *Cellular automata for decentralized control of self-reconfigurable robots*, ICRA 2001 Workshop on Modular Self-Reconfigurable Robots, (2001).

[8] H. CHEN AND D. DOTY, *Parallelism and time in hierarchical self-assembly*, in Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2012, pp. 1163–1182.

[9] G. CHIRIKJIAN, *Metamorphic hyper-redundant manipulators*, Proc. of Intl. Conf. on Advanced Mechatronics, (1993), pp. 467–472.

[10] R. M. DIRKS AND N. A. PIERCE, *Triggered amplification by hybridization chain reaction*, Proceedings of the National Academy of Sciences, 101 (2004), pp. 15275–15278.

[11] H. EHRIG, *Introduction to the algebraic theory of graph grammars (a survey)*, Lecture Notes in Computer Science, 73 (1979), pp. 1–69.

[12] F. FOSTER, M. ZHANG, A. DUCKETT, V. CUCEVIC, AND C. PAVLIN, *In vivo imaging of embryonic development in the mouse eye by ultrasound biomicroscopy*, Investigative ophthalmology & visual science, 44 (2003), p. 2361.

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

[13] S. C. GOLDSTEIN AND T. MOWRY, *Claytronics: A scalable basis for future robots*, Robosphere, Nov, (2004).
[14] S. J. GREEN, J. BATH, AND A. J. TURBERFIELD, *Coordinated chemomechanical cycles: A mechanism for autonomous molecular motion*, Physical Review Letters, 101 (2008), p. 238101.
[15] S. GRIFFITH, *Growing machines*, PhD thesis, MIT, 2004.
[16] R. GROSS AND M. DORIGO, *Self-assembly at the macroscopic scale*, Proceedings of the IEEE, 96 (2008), pp. 1490–1508.
[17] B. GRZYBOWSKI, C. WILMER, J. KIM, K. BROWNE, AND K. BISHOP, *Self-assembly: from crystals to cells*, Soft Matter, 5 (2009), pp. 1110–1128.
[18] H. HESS, *Self-assembly driven by molecular motors*, Soft Matter, 2 (2006), pp. 669–677.
[19] M. G. L. V. D. HEUVEL AND C. DEKKER, *Motor proteins at work for nanotechnology*, Science, 317 (2007), p. 333.
[20] C. JONES AND M. J. MATARIC, *From local to global behavior in intelligent self-assembly*, IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA?03, 1 (2003).
[21] E. KARSENTI, *Self-organization in cell biology: a brief history*, Nature Reviews Molecular Cell Biology, 9 (2008), pp. 255–262.
[22] E. R. KAY, D. A. LEIGH, AND F. ZERBETTO, *Synthetic molecular motors and mechanical machines*, Angew. Chem. Int. Ed., 46 (2007).
[23] E. KLAVINS, R. GHRIST, AND D. LIPSKY, *Graph grammars for self-assembling robotic systems*, in Proceedings of the International Conference on Robotics and Automation, 2004, pp. 5293–5300.
[24] A. LINDENMAYER, *Models for multicellular development: Characterization, inference and complexity of L-systems*, Trends, Techniques, and Problems in Theoretical Computer Science, (1987), pp. 138–168.
[25] A. LIU AND D. FLETCHER, *Biology under construction: in vitro reconstitution of cellular function*, Nature Reviews Molecular Cell Biology, 10 (2009), pp. 644–650.
[26] W. LU AND C. LIEBER, *Nanoelectronics from the bottom up*, Nature materials, 6 (2007), pp. 841–850.
[27] R. LUECKE, W. WOSILAIT, AND J. YOUNG, *Mathematical modeling of human embryonic and fetal growth rates*, Growth Development and Aging, 63 (1999), pp. 49–59.
[28] K. LUND, A. MANZO, N. DABBY, N. MICHELOTTI, A. JOHNSON-BUCK, J. NANGREAVE, S. TAYLOR, R. PEI, M. STOJANOVIC, N. WALTER, ET AL., *Molecular robots guided by prescriptive landscapes*, Nature, 465 (2010), pp. 206–210.
[29] D. MORRIS, M. GREALY, H. LEESE, AND G. R. CENTRE, *Cattle embryo growth, development and viability*, Grange Research Centre, 2001.
[30] S. MURATA, H. KUROKAWA, AND S. KOKAJI, *Self-assembling machine*, in Proceedings of the 1994 International Conference on Robotics and Automation, San Diego, CA, USA, 1994, pp. 441–448.
[31] R. MUSCAT, J. BATH, AND A. TURBERFIELD, *A programmable molecular robot*, Nano Lett, 11 (2011), pp. 982–987.
[32] R. NAGPAL, *Programmable pattern-formation and scale-independence*, Proc International Confernce on Complex Systems (ICCS), (2002).
[33] T. OMABEGHO, R. SHA, AND N. SEEMAN, *A bipedal DNA brownian motor with coordinated legs*, Science, (2009).
[34] R. PEI, S. TAYLOR, D. STEFANOVIC, S. RUDCHENKO, T. MITCHELL, AND M. STOJANOVIC, *Behavior of polycatalytic assemblies in a substrate-displaying matrix*, Journal of the American Chemical Society, 128 (2006), pp. 12693–12699.
[35] P. PRUSINKIEWICZ, *Modeling plant growth and development*, Current opinion in plant biology, 7 (2004), pp. 79–83.
[36] M. D. ROSA, S. GOLDSTEIN, P. LEE, J. CAMPBELL, AND P. PILLAI, *Scalable shape sculpting via hole motion: Motion planning in lattice-constrained modular robots*, Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA?06), (2006).
[37] D. RUS AND M. VONA, *Crystalline robots: Self-reconfiguration with compressible unit modules*, Autonomous Robots, 10 (2001), pp. 107–124.
[38] K. SAITOU, *Conformational switching in self-assembling mechanical systems*, IEEE Transactions on Robotics and Automation, 15 (1999), pp. 510–520.
[39] G. SEELIG, D. SOLOVEICHIK, D. ZHANG, AND E. WINFREE, *Enzyme-free nucleic acid logic circuits*, Science, 314 (2006), pp. 1585–1588.
[40] R. SIROMONEY, *Array languages and Lindenmayer systems-a survey, in" The Book of L" Eds. G. Rozenberg and A. Salomaa*, 1985.
[41] D. SOLOVEICHIK AND E. WINFREE, *Complexity of self-assembled shapes*, SIAM Journal on Computing, 36 (2007), pp. 1544–1569.
[42] S. STUPP, *Self-Assembly and Biomaterials*, Nano letters, (2010), pp. 249–255.
[43] Y. TIAN, Y. HE, Y. CHEN, P. YIN, AND C. MAO, *A DNAzyme that walks processively and autonomously along a one-dimensional track*, Angewandte Chemie International Edition, 44 (2005), pp. 4355–4358.
[44] S. VENKATARAMAN, R. DIRKS, P. ROTHEMUND, E. WINFREE, AND N. PIERCE, *An autonomous polymerization motor powered by DNA hybridization*, Nature Nanotechnology, 2 (2007), pp. 490–494.
[45] J. VENTER, M. ADAMS, E. MYERS, P. LI, R. MURAL, G. SUTTON, H. SMITH, M. YANDELL, C. EVANS, R. HOLT, ET AL., *The sequence of the human genome*, Science, 291 (2001), pp. 1304–1351.
[46] J. E. WALTER, J. L. WELCH, AND N. M. AMATO, *Distributed reconfiguration of metamorphic robot chains*, Distributed Computing, 17 (2004), pp. 171–189.
[47] J. WERFEL AND R. NAGPAL, *Towards a common comparison framework for global-to-local programming of self-assembling robotic systems*, IEEE Conference on Intelligent Robots and Systems, (2007).
[48] P. WHITE, K. KOPANSKI, AND H. LIPSON, *Stochas-*

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.

tic self-reconfigurable cellular robotics, IEEE International Conference on Robotics and Automation, 3 (2004), pp. 2888–2893.

[49] G. WHITESIDES AND B. GRZYBOWSKI, *Self-assembly at all scales*, Science, 295 (2002), p. 2418.

[50] M. N. WIN AND C. D. SMOLKE, *Higher-order cellular information processing with synthetic rna devices*, Science, 322 (2008), p. 456.

[51] E. WINFREE, *On the computational power of DNA annealing and ligation*, in DNA Based Computers, R. Lipton and E. Baum, eds., American Mathematical Society, Providence, RI, 1996, pp. 199–221.

[52] D. WOODS, H.-L. CHEN, S. GOODFRIEND, N. DABBY, E. WINFREE, AND P. YIN, *Efficient active self-assembly of shapes*. Manuscript in preparation.

[53] M. YIM, J. LAMPING, E. MAO, AND J. G. CHASE, *Rhombic dodecahedron shape for self-assembling robots*, SPL TechReport P9710777, Xerox PARC, (1997).

[54] M. YIM, W. SHEN, B. SALEMI, D. RUS, M. MOLL, H. LIPSON, E. KLAVINS, AND G. S. CHIRIKJIAN, *Modular self-reconfigurable robot systems*, IEEE Robotics and Automation Magazine, 14 (2007), p. 43.

[55] P. YIN, H. M. T. CHOI, C. R. CALVERT, AND N. A. PIERCE, *Programming biomolecular self-assembly pathways*, Nature, 451 (2008), pp. 318–322.

[56] P. YIN, H. YAN, X. DANIELL, A. J. TURBERFIELD, AND J. REIF, *A unidirectional DNA walker that moves autonomously along a track*, Angewandte Chemie International Edition, 43 (2004), pp. 4906–4911.

[57] B. YURKE, A. TURBERFIELD, A. MILLS, F. SIMMEL, AND J. NEUMANN, *A DNA-fuelled molecular machine made of DNA*, Nature, 406 (2000), pp. 605–608.

[58] D. ZHANG, A. TURBERFIELD, B. YURKE, AND E. WINFREE, *Engineering entropy-driven reactions and networks catalyzed by DNA*, Science, 318 (2007), pp. 1121–1125.

[59] D. ZHANG AND E. WINFREE, *Control of DNA strand displacement kinetics using toehold exchange*, J. Am. Chem. Soc, 131 (2009), pp. 17303–17314.

Copyright © SIAM.
Unauthorized reproduction of this article is prohibited.