DNA Origami Words and Rewriting Systems

James Garrett, Nataša Jonoska, Hwee Kim, and Masahico Saito

Department of Mathematics and Statistics, University of South Florida
4202 E. Fowler Ave., Tampa, FL 33620, USA
jgarrett1@mail.usf.edu, jonoska@mail.usf.edu,
hweekim@mail.usf.edu, saito@usf.edu

Abstract. We classify rectangular DNA origami structures according to their scaffold and staples organization by associating a graphical representation to each scaffold folding. Inspired by well studied Temperley-Lieb algebra, we identify basic modules that form the structures. The graphical description is obtained by 'gluing' basic modules one on top of the other. To each module we associate a symbol and every word corresponds to a graphical representation of a DNA origami structure. A set of rewriting rules defines equivalent words that correspond to the same graphical structure. We propose two different types of basic module structures and corresponding rewriting rules. For each type, we provide the number of all possible structures through the number of equivalence classes of words. We also give a polynomial time algorithim that gives the shortest word for each equivalence class.

1 Introduction

Self-assembly is a process where smaller components (usually molecules) autonomously assemble to form a larger structure. Self-assembly plays an important role in building biomolecular structures and high order polymers [16]. Applications of self-assembly include nanostructured electric circuits [1,5] and smart drug delivery [10,15]. A well-known variant of self-assembly is DNA origami introduced by Rothemund [12] where a single-stranded DNA plasmid, called the *scaffold*, outlines a shape, while short DNA strands, called *staples*, connect different parts of the scaffold, fixing the terminal rigid structure. The left side of Fig. 1 shows a segment of schematic DNA origami where the scaffold is depicted by a black line while staples are represented by colored lines with arrows. Experimental results of several DNA origami shapes from Rothemund's original paper [12] are shown to the right of Fig. 1.

Theoretical approaches to analyze DNA origami have been focused on efficient sequence design of staples as well as synthetic scaffolds that fold into the target shape [11,14]. However, the same outlined shape can be obtained in various different scaffold and staple organizations. In this paper, we use graphical description to describe different scaffold/staple organization within the same origami shape. We identify unit building blocks (modules) for the graphical representations whose composition (one on top of another) through connecting the

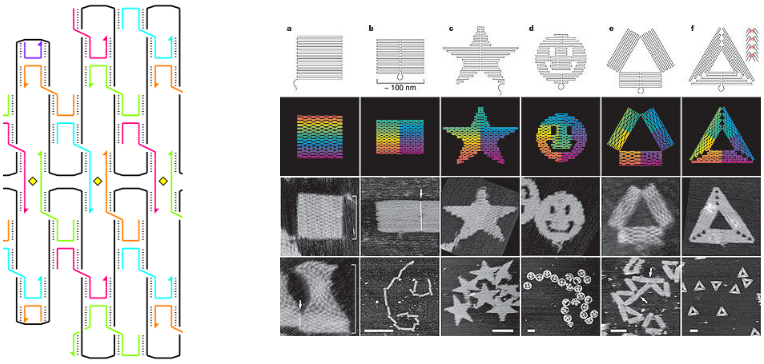


Fig. 1. (Left) A schematic representation of a DNA origami structure. The scaffold is a black line and staples are colored lines with arrows. (Right) Various shapes made by DNA origami. Both figures are from Rothemund [12].

corresponding staple/scaffold strands builds up larger structure. The unit blocks correspond to symbols in an alphabet, and concatenation of symbols correspond to composition of the modules. It can be observed that the unit structures within DNA origami are closely related to the diagram representation of the well studied Temperley-Lieb algebras. Inspired by these algebras and their monoidal variants (Jones and Kauffman monoids [3,7,8]), we define rewriting rules that provide equivalence of words corresponding to their graphical representation equivalence. In this way, the set of graphical representations of all possible DNA structures outlining a shape, correspond to the set of equivalence classes of words obtained through the rewriting rules. We propose two different types of basic module structures and their corresponding rewriting rules. For each type, we provide the number of distinct equivalence classes of the words, and hence of the possible DNA origami structures. We also compute the size of the maximum word within each class and provide a polynomial time algorithm to obtain the shortest length word within the class.

2 Preliminaries

An alphabet Σ is a non-empty finite set of symbols. A word $w = w_1 w_2 \cdots w_n \in \Sigma^n$ is a finite sequence of n symbols over Σ, and $|w| = n$ denotes the size of the word. We use ϵ to denote the empty word. A *subword* or a *factor* of a word $w = w_1 w_2 \cdots w_n$ is $w' = w_i \cdots w_j$ where $1 \leq i \leq j \leq n$. We use Σ^* to denote the set of all words over Σ. Concatenation of two words x and y is denoted by $x \cdot y$, or simply xy.

A word rewriting system (Σ, R) consists of an alphabet Σ and a set $R \subseteq \Sigma^* \times \Sigma^*$ of rewriting rules. An element (x, y) of R is called a rewriting rule, and is written as $x \to y$. In general, we may rewrite uxv as uyv for $u, v \in \Sigma^*$ if $(x, y) \in R$, and denote by $uxv \to uyv$. For a sequence of words $u = x_1 \to$

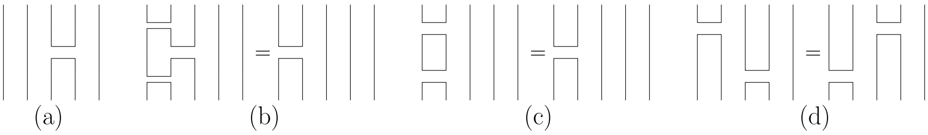


Fig. 2. Graphical representation of the Jones monoid $\mathcal{J}_4$. (a) The generator h_3 (b) The relation $h_1h_2h_1 = h_1$ (c) The relation $h_1h_1 = h_1$ (d) The relation $h_1h_3 = h_3h_1$

$x_2 \to \cdots \to x_n = v$ in a rewriting system (Σ, R), we write $u \to_* v$. We define an *equivalence class* of a word w as $[w] = \{w' \mid w' \to_* w \text{ using } R\}$. A word $w_0 \in [w]$ is *irreducible* if $|w_0| \leq |w'|$ for all $w' \in [w]$. We use the lexicographically first irreducible word $\hat{w}$ of $[w]$ as the *representative word* of $[w]$. We can define the set of distinct equivalence classes $\mathcal{O}$, and refer to an equivalence class in $\mathcal{O}$ by its representative word if the context is clear. The readers may refer to Book and Otto [2] for more information about word rewriting systems.

The Temperley-Lieb algebra TL_n has been extensively studied in physics and knot theory [8]. The monoid versions of Temperley-Lieb algebras, called Kauffman monoids and Jones monoids $\mathcal{J}_n$, have been also well studied [3,7,9]. The generators of $\mathcal{J}_n$ are $h_1, \ldots, h_{n-1}$ and satisfy three classes of relations:

1. $h_ih_jh_i = h_i$ for $|i - j| = 1$
2. $h_ih_i = h_i$
3. $h_ih_j = h_jh_i$ for $|i - j| \geq 2$

The generators and relations can be represented graphically as in Fig. 2 [9]. Each generator h_i in $\mathcal{J}_n$ is part of a structure that has $n + 1$ vertical lines such that a line connects the top (and bottom) ith and $i + 1$st endpoint. The generator h_3 in $\mathcal{J}_4$ is presented in Fig. 2 (a), connecting the 3rd and the 4th top and bottom points respectively. Multiplication of two elements corresponds to concatenation of diagrams, placing the diagram of the first element on top of the second, and removing closed loops. The relations 1 to 3 can also be expressed graphically as in Fig. 2 (b) to (d), respectively. Two elements in the Jones monoid are equal if their graphical representations are equivalent, that is, they have the same set of connecting segments except loops. For any two elements that have equivalent diagrams, one word can be rewritten to the other using the sequence of relations 1 to 3. In simplification of the DNA origami structure, we take the similar approach that we only take the account of scaffolds and staples that are visible at the borderline of the whole structure. Thus, we use the Jones monoid as a base to construct DNA origami words and rewriting systems.

3 DNA Origami Words and Rewriting Systems

3.1 DNA Origami Words

We focus on rectangular DNA origami structures. They can be formed through variety of ways to fold a scaffold strand and organize the staples connecting the scaffolds. We present an algebraic way to distinguish these different ways of obtaining the same overall shape. We take basic unit structures (modules) that construct the structure and associate symbols (generators) to these basic modules. Based on graphical diagrams, and inspired by the Jones monoid diagrams, we define equivalence of two origami structures. We define corresponding rewriting rules that realize the equivalence in the graphical diagrams.

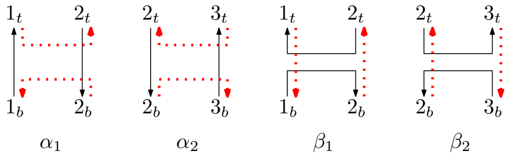


Fig. 3. Graphical representation of units of α_i and β_i. Scaffolds are represented by black lines and staples are represented by red dotted lines. For better visibility, staples are shifted right.

In the DNA origami structure, we observe that the structure has columns made of scaffolds, and staples go along the scaffolds. Between two columns, there are points where two adjacent scaffolds cross, and also points where two adjacent staples cross. In addition, scaffolds and staples have directions: adjacent scaffolds are anti-parallel, and a scaffold is anti-parallel to the staple on the scaffold. We represent a graphical structure with types of directed segments and the corresponding end-point connections. In addition, in order to define composition of structures when staples are missing in some parts of the structures, we consider 'virtual' staples. We use $p = i_t$ (i_b) to represent a point at the top (bottom) of the ith column. We assume that scaffolds at the ith column go downward if i is odd, and upward if i is even. Hence, a *graphical structure* is a tuple $(\mathcal{R}_{sca}, \mathcal{V}_{sca}, \mathcal{R}_{sta}, \mathcal{V}_{sta})$ of sets of ordered pairs (p, q) of points $p, q \in \{i_x \mid 1 \leq i \leq n, x = \text{t or b}\}$. The set $\mathcal{R}_{sca}$ $(\mathcal{R}_{sta})$ contains ordered pairs (p, q) of points, each pair representing a *real* scaffold (staple) starting from p and ending at q, respectively. The set $\mathcal{V}_{sca}$ $(\mathcal{V}_{sta})$ contains ordered pairs of points that represent *virtual* scaffolds (staples), which are not visible. Namely, for columns without scaffolds (staples), we assume that there exist straight scaffolds (staples) which are not visible, for convenience of definition of concatenation. For an ordered pair (p, q) of points, we define the reversal pair as (q, p). We define basic modules and corresponding generators, given n as the width of the structure. We use $\Sigma_n = \{\alpha_i, \beta_i \mid 1 \leq i \leq n-1\}$ as an alphabet for DNA origami words with the order $\alpha_1 < \cdots < \alpha_{n-1} < \beta_1 < \cdots < \beta_{n-1}$. For each generator α_i, β_i, Table 1 shows the set of pairs of scaffolds and staples that describe structures between the ith and the i+1st columns. The four pairs that describe α_i (resp. β_i) are called *units* for α_i (resp. β_i). The units of the generator α_i (β_i) are shown in Fig. 3.

Each generator $\gamma_i \in \Sigma_n$ has a *context* $\mathcal{C}(\gamma_i)$ which consists of pairs (k_t, k_b) and their reverses for $k \notin \{i, i+1\}$. The pairs in $\mathcal{C}(\gamma_i)$ can be real or virtual. Table 1

	$\mathcal{R}_{sca}$	$\mathcal{R}_{sta}$
α_i	$(i_b, i_t), (i{+}1_t, i{+}1_b)$	$(i_t, i{+}1_t), (i{+}1_b, i_b)$
β_i	$(i{+}1_t, i_t), (i_b, i{+}1_b)$	$(i_t, i_b), (i{+}1_b, i{+}1_t)$

Table 1. Units for generators of odd i's (pairs are reversed for even i's).

	$k \notin \{i, i+1\}$	$\mathcal{R}_{sca}$	$\mathcal{V}_{sca}$	$\mathcal{R}_{sta}$	$\mathcal{V}_{sta}$
$\mathcal{G}_{max(n)}$	odd k	(k_b, k_t)		(k_t, k_b)	
	even k	(k_t, k_b)		(k_b, k_t)	
$\mathcal{G}_{sta(n)}$	odd k	(k_b, k_t)			(k_t, k_b)
	even k	(k_t, k_b)			(k_b, k_t)
$\mathcal{G}_{min(n)}$	odd k		(k_b, k_t)		(k_t, k_b)
	even k		(k_t, k_b)		(k_b, k_t)

Table 2. Summary of the context for odd i's (pairs are reversed for even i's).

describes an example of three situations that can be used for three different descriptions of graphical structures $\mathcal{G}_{max(n)}$, $\mathcal{G}_{sta(n)}$, $\mathcal{G}_{min(n)}$, each representing γ_i.

We note that in $\mathcal{G}_{max(n)}$ the context $\mathcal{C}(\gamma_i)$ has both $\mathcal{V}_{sca}$ and $\mathcal{V}_{sta}$ empty, while in $\mathcal{G}_{sta(n)}$ the context $\mathcal{C}(\gamma_i)$ has $\mathcal{V}_{sca} = \emptyset$. In the case of $\mathcal{G}_{min(n)}$, all pairs of the context $\mathcal{C}(\gamma_i)$ are virtual. Graphical structures of α_2's in different $\mathcal{G}$'s are shown in Fig. 4.

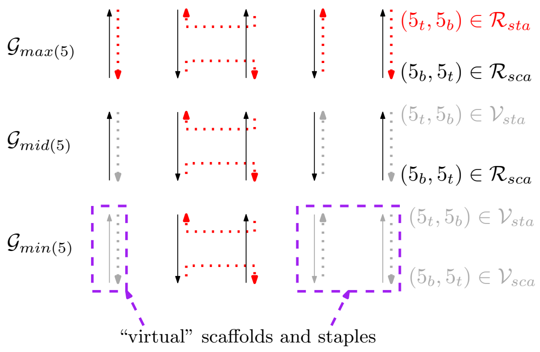

$\mathcal{G}_{max(5)}$
$(5_t, 5_b) \in \mathcal{R}_{sta}$
$(5_b, 5_t) \in \mathcal{R}_{sca}$
$\mathcal{G}_{mid(5)}$
$(5_t, 5_b) \in \mathcal{V}_{sta}$
$(5_b, 5_t) \in \mathcal{R}_{sca}$
$\mathcal{G}_{min(5)}$
$(5_t, 5_b) \in \mathcal{V}_{sta}$
$(5_b, 5_t) \in \mathcal{V}_{sca}$
"virtual" scaffolds and staples

Fig. 4. Different graphical structures of α_2's in $\mathcal{G}_{max(5)}$, $\mathcal{G}_{sta(5)}$ and $\mathcal{G}_{min(5)}$. Virtual scaffolds and staples are colored in gray.

Concatenation of words and the corresponding graphical structure is defined similarly as in the Jones monoid diagrams. Graphical structures that correspond to words in Σ_n^* are obtained by joining graphical structures of generators as explained below. Concatenating two words correspond to joining graphical structures. We place the graphical structure of the first word on top of the second and connect the vertical lines that meet. In the case of existence of virtual staples or scaffolds do the following: If a real scaffold (staple) meets a virtual scaffold (staple), then the virtual scaffold (staple) becomes real.

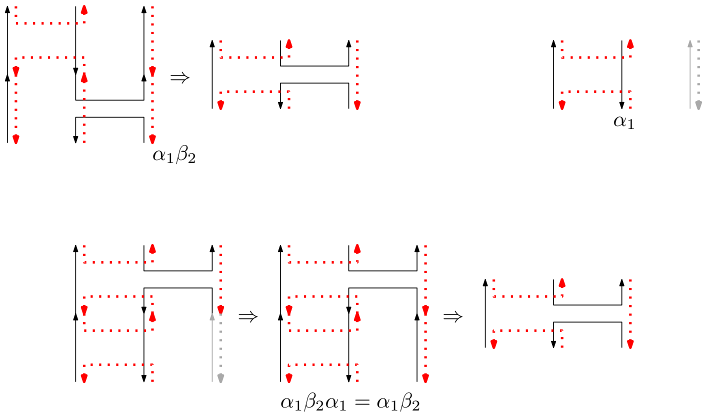


Fig. 5. Concatenation of $\alpha_1\beta_2$ and α_1 under $\mathcal{G}_{min(3)}$

Fig. 5 shows concatenation of $\alpha_1\beta_2$ and α_1 under $\mathcal{G}_{min(3)}$. Formally the graphical structure of a word $w = w_1w_2$ is defined as follows: Suppose we have two words w_1 and w_2 with graphical structures $G(w_1) = (\mathcal{R}_{sca1}, \mathcal{V}_{sca1}, \mathcal{R}_{sta1}, \mathcal{V}_{sta1})$ and $G(w_2) = (\mathcal{R}_{sca2}, \mathcal{V}_{sca2}, \mathcal{R}_{sta2}, \mathcal{V}_{sta2})$ respectively, then we obtain the graphical structure $G(w) = (\mathcal{R}_{sca}, \mathcal{V}_{sca}, \mathcal{R}_{sta}, \mathcal{V}_{sta})$ with the following: The scaffold sets ($\mathcal{R}_{sca}$ and $\mathcal{V}_{sca}$) are obtained with the procedure (the staples follow an equivalent procedure):

1. For all ordered pairs in $\mathcal{R}_{sca1}$ and $\mathcal{V}_{sca1}$, replace the subscript b by m.
2. For all ordered pairs in $\mathcal{R}_{sca2}$ and $\mathcal{V}_{sca2}$, replace the subscript t by m.
3. Let $\mathcal{R}_{sca} = \mathcal{R}_{sca1} \cup \mathcal{R}_{sca2}$ and $\mathcal{V}_{sca} = \mathcal{V}_{sca1} \cup \mathcal{V}_{sca2}$.
4. (Connecting scaffolds) Repeatedly find one of the following pairs of scaffolds if possible and do the corresponding process. Otherwise, move to the next step.
   (a) If there exist $(p, i_m), (i_m, q) \in \mathcal{R}_{sca}$, delete them and add (p, q) to $\mathcal{R}_{sca}$.
   (b) If there exist $(p, i_m) \in \mathcal{R}_{sca}$ and $(i_m, q) \in \mathcal{V}_{sca}$, delete them and add (p, q) to $\mathcal{R}_{sca}$.
   (c) If there exist $(p, i_m) \in \mathcal{V}_{sca}$ and $(i_m, q) \in \mathcal{R}_{sca}$, delete them and add (p, q) to $\mathcal{R}_{sca}$.
   (d) If there exist $(p, i_m), (i_m, q) \in \mathcal{V}_{sca}$, delete them and add (p, q) to $\mathcal{V}_{sca}$.
5. For every pair $(p, q) \in \mathcal{R}_{sca} \cup \mathcal{V}_{sca}$, delete it if p or q has subscript m.

Fig. 6 describes concatenation process of scaffolds. We replace the subscripts for the bottom points of w_1 and the top points of w_2 by m, which denotes the

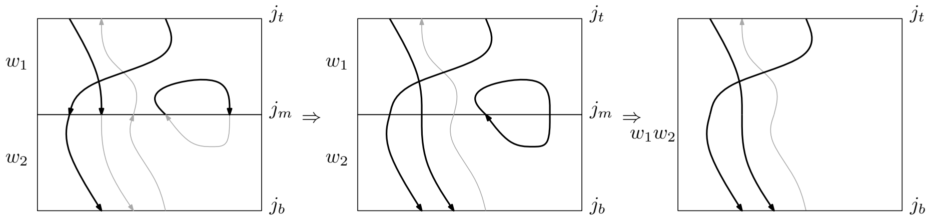


Fig. 6. Scaffold concatenation of w_1 and w_2. Real scaffolds are represented by thick lines for better visibility.

middle points. Then, we repeatedly search for pairs of scaffolds that meet at the middle and connect them. We regard the connected scaffold to be virtual only if both original scaffolds were virtual (step 4 (d)). Finally, we delete all pairs of scaffolds whose endpoints are at the middle, this includes all internal loops. Based on Tables 1 and 2, we can define $\mathcal{G}$'s as the sets of all graphical structures that can be constructed by concatenation of generators.

3.2 DNA Origami Rewriting Systems

It is straightforward that for any alphabet, given two words, the graphical structure of their concatenation is unique. Thus, if $G(w_1) = G(w_2)$ under a $\mathcal{G}$, we may say $w_1 \sim w_2$ under a $\mathcal{G}$ and define a rewriting rule $w_1 \leftrightarrow w_2$ between two equivalent words. Due to difference of context structures, rewriting rules for $\mathcal{G}_{max(n)}$, $\mathcal{G}_{sta(n)}$ and $\mathcal{G}_{min(n)}$ differ from each other. For each structure, we find the set of basic rewriting rules that is sufficient to describe equivalence, and analyze the set of distinct equivalence classes.

$\boldsymbol{\mathcal{G}_{max(n)}}$ Case We first observe that all staples and scaffolds in $\mathcal{G}_{max(n)}$ are real. We observe that staples in α_i (and scaffolds in β_i) are identical to the diagram of h_i in $\mathcal{J}_n$ except directions, and directions of scaffolds and staples do not make conflict in concatenation, which results in bijection between them. Moreover, scaffolds in α_i (and staples in β_i) are straight and do not affect the structure of scaffolds (staples) when concatenated. For convenience, we use γ and δ to represent an arbitrary generator, and $\overline{\gamma}$ to denote the complementary generator of γ. Then we have the inter-commutation rewriting rule $\gamma_i\overline{\gamma_j} \leftrightarrow \overline{\gamma_j}\gamma_i$ as in Fig. 7.

We may apply rewriting rules from $\mathcal{J}_n$ separately to w_a and w_b, resulting in the following rewriting rules:

1. (inter-commutation rule) $\gamma_i\overline{\gamma_j} \leftrightarrow \overline{\gamma_j}\gamma_i$
2. (idempotency rule) $\gamma_i\gamma_i \leftrightarrow \gamma_i$
3. (intra-commutation rule) $\gamma_i\gamma_j \leftrightarrow \gamma_j\gamma_i$ for $|i-j| \geq 2$
4. (TL relation rule) $\gamma_i\gamma_j\gamma_i \leftrightarrow \gamma_i$ for $|i-j| = 1$

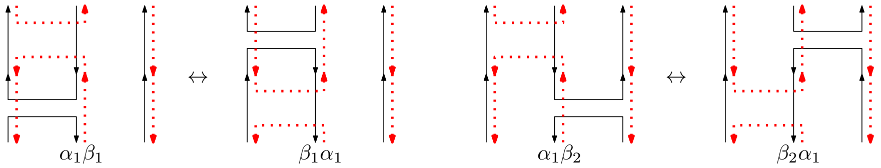


Fig. 7. Inter-commutation rewriting rule for $\mathcal{G}_{max(3)}$

Based on the set $R_{max(n)}$ of these rules, we can define the set $\mathcal{O}_{max(n)}$ of distinct equivalence classes. We say that a rule is *non-increasing* if the resulting word is not longer than the original word. In the above rules, rules 1,3 and right directions of rules 2 and 4 form the set of non-increasing rules. We call a sequence of rewriting rules with only non-increasing rules as non-increasing rewriting.

Let $\Sigma_{(\alpha)n} = \{\alpha_1, \ldots, \alpha_{n-1}\}$ and $\Sigma_{(\beta)n} = \{\beta_1, \ldots, \beta_{n-1}\}$. Using the inter-commutation rule, we may rewrite any word w to $w_a w_b$, where $w_a \in \Sigma^*_{(\alpha)n}$ and $w_b \in \Sigma^*_{(\beta)n}$. We say that such word $w_a w_b$ is in an inter-commutation-free form. Also, using intra-commutation rules, we may set additional conditions for $w_a = (\alpha_{i_1}\alpha_{i_1-1}\cdots\alpha_{k_1})(\alpha_{i_2}\alpha_{i_2-1}\cdots\alpha_{k_2})\cdots(\alpha_{i_p}\alpha_{i_p-1}\cdots\alpha_{k_p})$ where i_p is the maximum subscript in w_a, $i_{j+1} > i_j$ and $k_{j+1} > k_j$ for $1 \le k < p$, and similar condition for w_b. Such w_a and w_b are unique [7], and we call such $w_a w_b$ as a commutation-free form of w.

We may regard the graphical structure of a word as a pair of scaffolds and staples, which can be regarded as two independent Jones monoid diagrams. Knowing that the relations 1 to 3 of the Jones monoid can sufficiently describe equivalence of diagrams, we have the following theorem:

Theorem 1. *For all $w_1, w_2 \in \Sigma^*_n$, $G(w_1) = G(w_2)$ under $\mathcal{G}_{max(n)}$ if and only if $w_1 \to_* w_2$ using $R_{max(n)}$. In other words, there exists bijection between $\mathcal{G}_{max(n)}$ and $\mathcal{O}_{max(n)}$.*

Given n, the number of elements of $\mathcal{J}_n$ is equal to the Catalan number $C_n = \dfrac{1}{n+1}\dbinom{2n}{n}$ [9], and the maximum size of the element is $\left\lfloor \dfrac{n^2}{4} \right\rfloor$ [4,7]. Thus, the following remark holds.

Remark 1. Given n, $|\mathcal{O}_{max(n)}| = \left(\dfrac{1}{n+1}\dbinom{2n}{n}\right)^2$, and the maximum size of a representative word in $\mathcal{O}_{max(n)}$ is $2\left\lfloor \dfrac{n^2}{4} \right\rfloor$.

We may regard relations 1 to 3 of the Jones monoid as rewriting rules, and define the set $\mathcal{O}_{j(n)}$ of distinct equivalence classes. Since the graphical structures in $\mathcal{G}_{max(n)}$ correspond to products of two Jones monoid diagrams, we first prove the following Lemma about the Jones monoid and use it to prove correctness of the proposed optimization algorithm.

Lemma 1. *Given two elements $w_1, w_2 \in \mathcal{J}_n$ where $w_2 \in [w_1]$ is irreducible, we can non-increasingly rewrite w_1 as w_2.*

Proof. The detailed proof is given in Appendix A. □

Theorem 2. *Given a word $w_0 \in [w_0] \in \mathcal{O}_{max(n)}$ of size m, we can find an irreducible word of $[w_0]$ within $O(nm^2)$ time.*

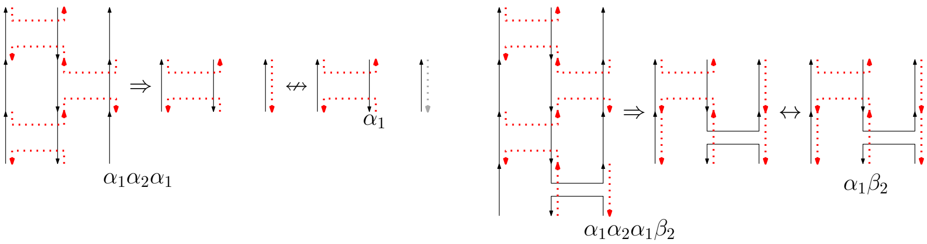
Proof. For given w_0, we first rewrite w_0 as w in the inter-commutation-free form, which takes $O(m^2)$ time. Then we repeatedly find one of the following conditions in w if possible and rewrite w accordingly:

1. If $w = v_1\gamma_i v_2\gamma_i v_3$ where $v_1, v_2, v_3 \in \Sigma_n^*$ and v_2 does not have γ_{i+1}, γ_i and γ_{i-1}, then rewrite w as $v_1 v_2 \gamma_i v_3$.
2. If $w = v_1\gamma_i v_2\gamma_j v_3\gamma_i v_4$ where $v_1, v_2, v_3, v_4 \in \Sigma_n^*$, $|i-j| = 1$ and v_2, v_3 do not have γ_{i+1}, γ_i and γ_{i-1}, then rewrite w as $v_1 v_2 \gamma_i v_3 v_4$.

It is straightforward that two rewritings can be done non-increasingly. We can iterate i's in the condition 1 and $(i, j = i+1$ or $i-1)$'s in the condition 2 to check if w satisfies the given form for given i in $O(m)$ time. It takes $O(nm)$ time to do one rewriting in 1 or 2, which decreases the size of the word by a constant. Thus, it takes $O(nm^2)$ to finish the whole process. If we consider the final word w', conditions in 1 and 2 are no longer satisfied and there is no sequence of non-increasing rules that decreases the size of the word. Then, w' becomes irreducible from Lemma 1. □

$\mathcal{G}_{sta(n)}$ Case Similar to the $\mathcal{G}_{max(n)}$ case, we have the following rewriting rules:

1. (Inter-commutation) $\gamma_i\overline{\gamma_j} \leftrightarrow \overline{\gamma_j}\gamma_i$
2. (Idempotency) $\gamma_i\gamma_i \leftrightarrow \gamma_i$
3. (Intra-commutation) $\gamma_i\gamma_j \leftrightarrow \gamma_j\gamma_i$ for $|i-j| \geq 2$



(a) (b)

Fig. 8. Examples of equivalence in $\mathcal{G}_{sta(3)}$. (a) $\alpha_1\alpha_2\alpha_1 \nsim \alpha_1$ (b) $\alpha_1\alpha_2\alpha_1\beta_2 \sim \alpha_1\beta_2$

Due to the lack of default real staples in generators, we cannot directly introduce the rewriting rule $\gamma_i\gamma_j\gamma_i \leftrightarrow \gamma_i$ for $|i-j| = 1$. For example, we cannot rewrite $\alpha_1\alpha_2\alpha_1$ as α_1, since $\alpha_1\alpha_2\alpha_1$ has a straight real staple at the third column while α_1 does not (See Fig. 8 (a).). We introduce the *span* of a word w as the set $span(w) = \bigcup_{\gamma_j \text{ in } w} \{j, j+1\}$ of columns.

Lemma 2. *Under $\mathcal{G}_{sta(n)}$, the span equals to the set of columns where real staples exist.*

Proof. The detailed proof is given in Appendix B. □

It is straightforward that two equivalent words should have the same span, and rewriting rules in the Jones monoid can be applied when both sides have the same span. For example, $\alpha_1\alpha_2\alpha_1\beta_2 \leftrightarrow \alpha_1\beta_2$ holds as in Fig. 8 (b). In general, we have the following additional rewriting rules, where $\delta \in \{\alpha, \beta\}$ and $v \in \Sigma_n^*$:

4. $\delta_j v\gamma_i\gamma_{i-1}\gamma_i \leftrightarrow \delta_j v\gamma_i$ if $j = i-1$ or $i-2$
5. $\delta_j v\gamma_i\gamma_{i+1}\gamma_i \leftrightarrow \delta_j v\gamma_i$ if $j = i+1$ or $i+2$
6. $\gamma_i\gamma_{i-1}\gamma_i v\delta_j \leftrightarrow \gamma_i v\delta_j$ if $j = i-1$ or $i-2$
7. $\gamma_i\gamma_{i+1}\gamma_i v\delta_j \leftrightarrow \gamma_i v\delta_j$ if $j = i+1$ or $i+2$

Rules 4 to 7 can be rewritten using rules 1 to 3 and the subset of rules 4 to 7. Before finding such subset, we define a *zig-zag word* $w \in \Sigma_n^*$ to be a word where each pair of adjacent generators in w have adjacent indices. We call a maximal subword of increasing (decreasing) indices as zig (zag). For example, $w = \alpha_3\alpha_4\alpha_3\alpha_2\alpha_1\alpha_2$ is a zig-zag word with a zig-zag-zig sequence in the word. Using rules 2 to 7, we can rewrite any zig-zag word that consists of single generators type γ as a zig-zag word with at most three zigs or zags, which we call the *zig-zag normal form.*

Theorem 3. *For $w_1 = \delta_j v\gamma_i\gamma_{i-1}\gamma_i$ and $w_2 = \delta_j v\gamma_i$ where $j = i-1$ or $i-2$, we can rewrite w_1 as w_2 using rules 1 to 3 and the following rule: $\gamma_j v\gamma_i\gamma_{i-1}\gamma_i \leftrightarrow \gamma_j v\gamma_i$ where $v \in \Sigma_{(\gamma)n}^*$ and $v = \epsilon$ or $\gamma_j v\gamma_i$ is in the zig-zag normal form.*

Proof. The detailed proof is given in Appendix C. □

Similar simplification works for rules 4 to 7, and we may use the following rewriting rules for $v \in \Sigma_{(\gamma)n}^*$ and $v = \epsilon$ or $\gamma_j v\gamma_i$ in rule 4 and 5 ($\gamma_i v\gamma_j$ in rule 6 and 7) is in the zig-zag normal form:

4. $\gamma_j v\gamma_i\gamma_{i-1}\gamma_i \leftrightarrow \gamma_j v\gamma_i$ if $j = i-1$ or $i-2$
5. $\gamma_j v\gamma_i\gamma_{i+1}\gamma_i \leftrightarrow \gamma_j v\gamma_i$ if $j = i+1$ or $i+2$
6. $\gamma_i\gamma_{i-1}\gamma_i v\gamma_j \leftrightarrow \gamma_i v\gamma_j$ if $j = i-1$ or $i-2$
7. $\gamma_i\gamma_{i+1}\gamma_i v\gamma_j \leftrightarrow \gamma_i v\gamma_j$ if $j = i+1$ or $i+2$

For given n, let the set $R_{sta(n)}$ of rewriting rules consist of the above seven kinds of rules for $1 \leq i, j \leq n$. We can define the rewriting system $\mathcal{O}_{sta(n)} = (\Sigma_n, R_{sta(n)})$.

Theorem 4. *For all $w_1, w_2 \in \Sigma_n^*$, $G(w_1) = G(w_2)$ under $\mathcal{G}_{sta(n)}$ if and only if $w_1 \rightarrow_* w_2$ using $R_{sta(n)}$. In other words, there exists bijection between $\mathcal{G}_{sta(n)}$ and $\mathcal{O}_{sta(n)}$.*

Proof. The detailed proof is given in Appendix D. □

We compute the number of equivalence classes of words in $|\mathcal{O}_{sta(n)}|$. We use a binary string of length n to represent the graphical structures in $\mathcal{G}_{sta(n)}$ such that the ith bit equals 1 if and only if the ith staple and the the ith scaffold is a straight line. Each binary string is uniquely determined with a tuple $(a_1, b_1, \ldots, a_k, b_k)$ where a_i (b_i) represents the number of ith consecutive 0's (1's). For example, the 8-bit binary string 00111000 corresponds to a tuple $(2, 3, 3, 0)$. In particular, the bit 0 corresponds to $(1, 0)$ and the bit 1 to $(0, 1)$. The set of tuples corresponding to binary strings of length n is denoted $T_n = \{p = (a_1, b_1, \ldots, a_k, b_k) \mid k \geq 1, \text{ for all i}, a_i, b_i \in \mathbb{N}^0 \text{ and } \sum_{i=1}^{k}(a_i + b_i) = n\}$.

Theorem 5. *Given $n \in \mathbb{N}^0$, for each tuple $p \in \bigcup_{1\leq i\leq n} T_i$, let $D(p) \in \mathbb{N}^0$ be recursively defined as follows:*

- *for $p \in T_0$, $D(0, 0) = 1$.*
- *for $p \in T_1$, $D(p) = 1$ if $p = (0, 1)$ and $D(p) = 0$ if $p = (1, 0)$.*
- *for $p = (a_1, b_1, \ldots, a_k, b_k) \in T_n$, $(n > 0)$ we have $D(p) = \prod_{i=1}^{k} D(a_i, 0)$.*
- *for $n > 1$, we have $D(0, n) = 1$ and $D(n, 0) = \left(\frac{1}{n+1}\binom{2n}{n}\right)^2 - \sum_{p\in T_n\setminus\{(n,0)\}} D(p)$.*

Then, $|\mathcal{O}_{sta(n)}|$ is given as

$$|\mathcal{O}_{sta(n)}| = d(n) = \sum_{p\in T_n} [D(p) \times x(p)] - n.$$

where

$$x(a_1, b_1, \ldots, a_k, b_k) = \begin{cases} (b_1 + 1) & \text{if } k = 1, \\ (b_1 + 1) \cdot \prod_{i=2}^{k-1} \left(\frac{b_i(b_i + 1)}{2} + 1\right) \cdot (b_k + 1) & \text{if } k \neq 1, a_1 = 0, \\ \prod_{i=1}^{k-1} \left(\frac{b_i(b_i + 1)}{2} + 1\right) \cdot (b_k + 1) & \text{if } k \neq 1, a_1 > 0. \end{cases}$$

Proof. The detailed proof is given in Appendix E. □

The sequence $d(n)$ for $1 \leq n \leq 10$ is $1, 4, 31, 253, 2247, 21817, 227326, 2499598, 28660639, 339816259$. It is not listed in the OEIS [13] list of sequences, and the non-recursive formula of $d(n)$ is still open.

Theorem 6. *An upper bound of the size $u(n)$ of a maximum representative word in $\mathcal{O}_{sta(n)}$ is given by $2^n - 2$.*

Proof. The detailed proof is given in Appendix F. □

The bound in Theorem 6 is not tight, and the exact size of a maximum irreducible word is open.

Theorem 7. *Given a word $w_0 \in [w_0] \in \mathcal{O}_{sta(n)}$ of size m, we can find an irreducible word of $[w_0]$ within $O(nm^2)$ time.*

The proofs of Lemma 1 and Theorem 2 work similarly for Theorem 7. We first rewrite w_0 as w in the inter-commutation-free form. Then we repeatedly find one of the following conditions in w if possible and rewrite w accordingly:

1. If $w = v_1\gamma_i v_2 \gamma_i v_3$ where $v_1, v_2, v_3 \in \Sigma_n^*$ and v_2 does not have γ_{i+1}, γ_i and γ_{i-1}, then rewrite w as $v_1 v_2 \gamma_i v_3$.
2. If $w = v_1\gamma_i v_2 \gamma_{i+1} v_3 \gamma_i v_4$ where $v_1, v_2, v_3, v_4 \in \Sigma_n^*$, v_2, v_3 do not have γ_{i+1}, γ_i and γ_{i-1}, there exists δ_{i+1} in v_1 or v_4, or δ_{i+2} in v_1, v_2, v_3 or v_4, then rewrite w as $v_1 v_2 \gamma_i v_3 v_4$.
3. If $w = v_1\gamma_i v_2 \gamma_{i-1} v_3 \gamma_i v_4$ where $v_1, v_2, v_3, v_4 \in \Sigma_n^*$, v_2, v_3 do not have γ_{i+1}, γ_i and γ_{i-1}, there exists δ_{i-1} in v_1 or v_4, or δ_{i-2} in v_1, v_2, v_3 or v_4, then rewrite w as $v_1 v_2 \gamma_i v_3 v_4$.

3.3 Concluding Remarks

We have proposed modules and corresponding generators for DNA origami structures, defined concatenation of words and rewriting rules, and analyzed equivalence classes based on graphical equivalence. One model that we have not discussed is $\mathcal{G}_{min(n)}$. For $\mathcal{G}_{min(n)}$, seven types of rewriting rules for $\mathcal{G}_{sta(n)}$ hold. Moreover, we may prove that Theorem 4 holds for $\mathcal{G}_{min(n)}$ using the similar proof. It turns out that there is bijection between $\mathcal{G}_{sta(n)}$ and $\mathcal{G}_{min(n)}$, and $\mathcal{O}_{sta(n)} = \mathcal{O}_{min(n)}$.

Graphical structures corresponding to generators α_i's and β_i's in Fig. 3 describe crossing of scaffolds and staples in DNA origami well, while using only two types of generators. Here we explore possible further development of generators that are more plausible to DNA origami.

The first observation on the current generators is that they are vertically and horizontally symmetric (without directions), which causes the graphical structure to always have a cup-shaped fragment of a real scaffold at the top as in Fig. 9 (a). DNA origami does not have such fragments at the border of the structure, which leads us to revise generators to define such borders. Fig. 9 (b) proposes four different generators that are used to substitute α_1. In these generators, we introduce asymmetric structures that can be used to construct borders of the structure. We may define generators for β similarly. Under the assumption that we use the same concatenation procedure, for a graphical structure that corresponds to α_1, we can make arbitrary number of scaffolds and staples virtual by concatenation of four new generators as in Fig. 9 (c). Now, suppose we define the rewriting system based on equivalence under such generators. For each pair of diagrams of $\mathcal{J}_n$, we have $2n$ staples and scaffolds which can become virtual. From analysis similar to the proof of Theorem 5, the size of the set of equivalence classes becomes $\left(\left(\frac{1}{n+1}\binom{2n}{n}\right)^2 - 1\right) \cdot 2^{2n} + 1$. The set of rewriting rules that are sufficient to describe equivalence under such generators is open.

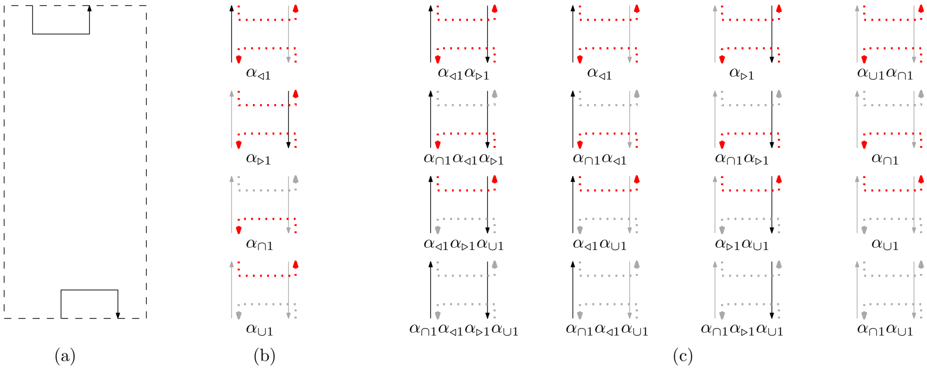

$\alpha_{\triangleleft 1}$
$\alpha_{\triangleright 1}$
$\alpha_{\cap 1}$
$\alpha_{\cup 1}$
$\alpha_{\triangleleft 1}\alpha_{\triangleright 1}$
$\alpha_{\triangleleft 1}$
$\alpha_{\triangleright 1}$
$\alpha_{\cup 1}\alpha_{\cap 1}$
$\alpha_{\cap 1}\alpha_{\triangleleft 1}\alpha_{\triangleright 1}$
$\alpha_{\cap 1}\alpha_{\triangleleft 1}$
$\alpha_{\cap 1}\alpha_{\triangleright 1}$
$\alpha_{\cap 1}$
$\alpha_{\triangleleft 1}\alpha_{\triangleright 1}\alpha_{\cup 1}$
$\alpha_{\triangleleft 1}\alpha_{\cup 1}$
$\alpha_{\triangleright 1}\alpha_{\cup 1}$
$\alpha_{\cup 1}$
$\alpha_{\cap 1}\alpha_{\triangleleft 1}\alpha_{\triangleright 1}\alpha_{\cup 1}$
$\alpha_{\cap 1}\alpha_{\triangleleft 1}\alpha_{\cup 1}$
$\alpha_{\cap 1}\alpha_{\triangleright 1}\alpha_{\cup 1}$
$\alpha_{\cap 1}\alpha_{\cup 1}$
(a)
(b)
(c)

Fig. 9. (a) In a graphical structure generated from α_i's and β_i's, there always exists a cup-shaped fragment of a real scaffold at the top (and cap-shaped fragment at the bottom). (b) Four revised generators that substitutes α_1. Virtual scaffolds and staples are colored in gray. (c) We can make arbitrary staples and scaffolds in α_1 virtual.

The second observation on the current generators is that we do not consider which side of the scaffold the staple is on. In the DNA origami structure, staples can be on the left or the right of the scaffold, and these two cases are distinguished. Moreover, for two adjacent staples at the opposite side of the same scaffold, they either disconnect or connect by crossing the scaffold. To model this observation, we may introduce revised graphical structures for α and β as in Fig. 10 (a). Staples are either at the left or the right of the scaffold, and some staple ends are extending which can be connected to other staples regardless of the side. We assume that two adjacent staples can be connected except when two are non-extending ends and at the opposite side. This additional condition for staple connection changes some of the commutation rewriting rules—for example, $\alpha_1\beta_1 \not\leftrightarrow \beta_1\alpha_1$ as in Fig. 10 (b). Algebraic analysis on relations based on such generators is done by Garrett et al. [6]. The set of rewriting rules that are sufficient to describe equivalence under such generators is still open.

Acknowledgment

This work is partially supported by NIH R01GM109459, and by NSF's CCF-1526485 and DMS-1800443. This research was also partially supported by the Southeast Center for Mathematics and Biology, an NSF-Simons Research Center for Mathematics of Complex Biological Systems, under National Science Foundation Grant No. DMS-1764406 and Simons Foundation Grant No. 594594.

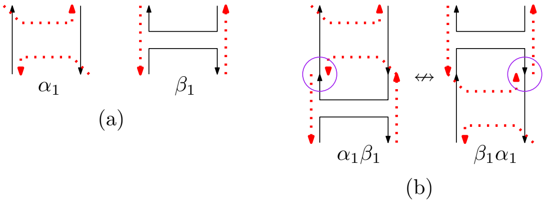


Fig. 10. (a) Revised generators α_1 and β_1. Two diagonal ends of staples in α_1 represent extending staple-ends. (b) Adjacent staples can be connected except when two are non-extending ends and at the opposite side.

References

1. T. Bhuvana, K. C. Smith, T. S. Fisher, and G. U. Kulkarni. Self-assembled CNT circuits with ohmic contacts using Pd hexadecanethiolate as in situ solder. *Nanoscale*, 1(2):271–275, 2009.
2. R. V. Book and F. Otto. *String-rewriting Systems*. Springer, 1993.
3. M. Borisavljević, K. Došen, and Z. Petric. Kauffman monoids. *Journal of Knot Theory and its Ramifications*, 11(2):127–143, 2002.
4. I. Dolinka and J. East. The idempotent-generated subsemigroup of the Kauffman monoid. *Glasgow Mathematical Journal*, 59(3):673–683, 2017.
5. Y. Eichen, E. Braun, U. Sivan, and G. Ben-Yoseph. Self-assembly of nanoelectronic components and circuits using biological templates. *Acta Polymerica*, 49(10-11):663–670, 1998.
6. J. Garrett, N. Jonoska, H. Kim, and M. Saito. Algebraic systems for DNA origami motivated from Temperley-Lieb algebras. *CoRR*, abs/1901.09120, 2019.
7. V. F. R. Jones. Index for subfactors. *Inventiones Matheematicae*, 72:1–25, 1983.
8. L. H. Kauffman. *Knots and Physics*. World Scientific, 2001.
9. K. W. Lau and D. G. FitzGerald. Ideal structure of the Kauffman and related monoids. *Communications in Algebra*, 34(7):2617–2629, 2006.
10. J. Li, C. Fan, H. Pei, J. Shi, and Q. Huang. Smart drug delivery nanocarriers with self-assembled DNA nanostructures. *Advanced Materials*, 25(32):4386–4396, 2013.
11. P. W. K. Rothemund. Design of DNA origami. In *Proceedings of 2005 International Conference on Computer-Aided Design*, pages 471–478, 2005.
12. P. W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006.
13. The on-line encyclopedia of integer sequences. `https://oeis.org/`.
14. R. Veneziano, S. Ratanalert, K. Zhang, F. Zhang, H. Yan, W. Chiu, and M. Bathe. Designer nanoscale DNA assemblies programmed from the top down. *Science*, 352(6293):1534, 2016.
15. G. Verma and P. A. Hassan. Self assembled materials: design strategies and drug delivery perspectives. *Physical Chemistry Chemical Physics*, 15(40):17016–17028, 2013.
16. G. M. Whitesides and M. Boncheva. Beyond molecules: Self-assembly of mesoscopic and macroscopic components. *Proceedings of the National Academy of Sciences of the United States of America*, 99(8):4769–4774, 2002.

A Proof of Lemma 1

We recall the rewriting rules for $\mathcal{J}_n$.

1. $h_i h_i \leftrightarrow h_i$
2. $h_i h_j \leftrightarrow h_j h_i$ for $|i-j| \geq 2$
3. $h_i h_j h_i \leftrightarrow h_i$ for $|i-j| = 1$

We rewrite the statement as follows: Suppose $w_1, w_2 \in \mathcal{J}_n$, w_2 is irreducible and we can rewrite w_1 as w_2 using t rules. Then, for all $t \geq 1$, there exists a sequence of non-increasing rules that rewrites w_1 as w_2. We prove the statement by induction on t.

– When $t = 1$, we have w_1 which can be rewritten as w_2 using one rule. Since w_2 is irreducible, the rule should be non-increasing.
– Suppose the statement holds for all $t \leq x$. When $t = x + 1$, we have the sequence of $x + 1$ rewriting rules that rewrites w_1 as w_2. If the first rule is non-increasing, then we can rewrite w_1 as w_2 by a sequence of non-increasing rules using the induction hypothesis. Now, we assume that the first rule is increasing and the rest of the rules are non-increasing. Note that rule 2 is the only rule that changes location of generators. Moreover, applying rules 1 or 3 on a word do not result in an additional pair of generators that rule 2 can be applied.
  - If the first rule is rule 1 ($h_i \to h_i h_i$), the resulting two α_i's should be involved in non-increasing rule 1 or 3 in the following sequence. The straight sequence of Fig. 11 (a) shows an example of such sequence where the left h_i is used in rule 1 and the right h_i is used in rule 3, where blue areas represent generators that h_i can switch the location with using rule 2. For such sequence, we can always find another sequence without increasing rules as the right sequence of Fig. 11 (a).
  - If the first rule is rule 3 ($h_i \to h_i h_j h_i$ for $|i-j| = 1$), the resulting h_j should be involved in non-increasing rule 1 or 3 in the following sequence. Without loss of generality, we assume that $j = i + 1$. Since $h_{i+1} h_i \neq h_i h_{i+1}$, h_j cannot be used in rule 1. The straight sequence of Fig. 11 (b) shows an example of such sequence where the middle $h_j = h_{i+1}$ is used in rule 3, and blue areas represent generators that h_{i+2} can switch the location with. For such sequence, we can always find another sequence without increasing rules as the right sequence of Fig. 11 (b).

B Proof of Lemma 2

We can prove the statement by induction on the size of the word. It is straightforward that the statement holds for generators. Assume that the statement holds for all $|w| = m$. For a word $w' = w\gamma_i$, if i and $i + 1$ are both in *span*(w), concatenation does not change the span and the statement holds. If i or $i + 1$ are not in *span*(w), then the span has new columns, and the graphical structure of w' has real staples for these columns, which makes the statement true.

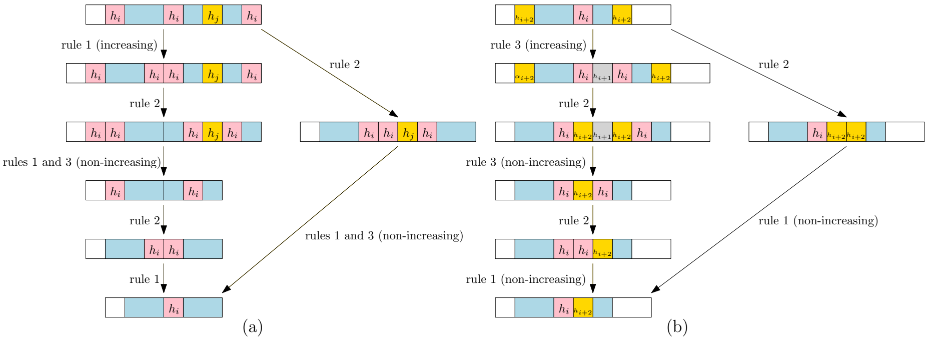


Fig. 11. Examples of sequences of rewriting rules that leads to the same element, where one uses only non-increasing rules. (a) The first rule is rule 1. (b) The first rule is rule 3.

C Proof of Theorem 3

First, we prove that we may assume $\delta = \gamma$. If $\delta = \overline{\gamma}$, we can rewrite v as a concatenation of a prefix $v_\gamma \in \Sigma^*_{(\gamma)n}$ and a suffix $v_\delta \in \Sigma^*_{(\delta)n}$. Then, $\delta_j v_\gamma v_\delta \gamma_i \gamma_{i-1} \gamma_i \to v_\gamma \delta_j \gamma_i \gamma_{i-1} \gamma_i v_\delta \to v_\gamma \delta_j \gamma_i v_\delta \to \delta_j v_\gamma v_\delta \gamma_i = \delta_j v \gamma_i$, and vice versa.

Second, we prove that we may assume $v \in \Sigma_{(\gamma)n}$. If $v \notin \Sigma_{(\gamma)n}$, we can rewrite v as a concatenation of a prefix $v_\gamma \in \Sigma^*_{(\gamma)n}$ and a suffix $v_{\overline{\gamma}} \in \Sigma^*_{(\overline{\gamma})n}$. Then, $\gamma_j v_\gamma v_{\overline{\gamma}} \gamma_i \gamma_{i-1} \gamma_i \to v_{\overline{\gamma}} \gamma_j v_\gamma \gamma_i \gamma_{i-1} \gamma_i \to v_{\overline{\gamma}} \gamma_j v_\gamma \gamma_i \to \gamma_j v_\gamma v_{\overline{\gamma}} \gamma_i = \gamma_j v \gamma_i$, and vice versa.

Now, given $\delta = \gamma$ and $v \in \Sigma_{(\gamma)n}$, we claim that we may assume $v = \epsilon$ or $\delta_j v \gamma_i$ is in a zig-zag normal form. For rule 4, We prove the statement by induction on the size of the maximal prefix p of v that is a zig-zag word. When $p = \epsilon$, if $v \neq \epsilon$, for the first generator γ_k of v, $k \neq j-1, j+1$ holds and we can switch δ_j and γ_k when $j \neq k$ or use rule 2 when $j = k$. In both cases, we can induce rule 4 using rules with $v = \epsilon$. Now assume that the statement holds for all $p \in \Sigma_{(\gamma)n}$ where $|p| \leq m$. For $|p| = m+1$, we may assume that $\delta_j p$ is in a zig-zag normal form, since rewriting to a zig-zag normal form can be done by rules in the induction hypothesis. We define the max index $max(p)$ (min index $min(p)$) of $\delta_j p$ to be the maximum (minimum) index of generators in $\delta_j p$. The m+2nd generator γ_t of v can switch with the last generator of p, and we have the following cases:

1. If $t > max(p) + 1$, we can move γ_t to the left of δ_j and the statement holds.
2. If $t = max(p)+1$, we consider two cases. If $\delta_j p$ has only a zig, then γ_t extends the zig and the statement holds. If $\delta_j p$ has a zig and a zag, then there exists a subword $\gamma_{max(p)-1}\gamma_{max(p)}\gamma_{max(p)-1}$ in p. We can rewrite the subword as $\gamma_{max(p)-1}$ using rules in the induction hypothesis, and then move γ_t to the left of δ_j.
3. If $min(p) \leq t \leq max(p)$, we can move γ_t into δ_j so that δ_j has a subword $\gamma_t \gamma_s \gamma_t$ where $|s-t| = 1$. We can rewrite the subword as γ_t using rules in the induction hypothesis.

4. If $t = min(p) + 1$, the case is similar to the case of $t = max(p) + 1$.
5. If $t < min(p) + 1$, we can move γ_t to the left of δ_j and the statement holds.

The above case analysis also holds for $p = v$, and we can claim that $\delta_j v \gamma_i$ is in a zig-zag normal form.

D Proof of Theorem 4

Note that if $w_1 \to_* w_2$ using $R_{sta(n)}$, then $G(w_1) = G(w_2)$ under $\mathcal{G}_{sta(n)}$ from definitions of rewriting rules. If $G(w_1) = G(w_2)$ under $\mathcal{G}_{max(n)}$ and $span(w_1) = span(w_2)$, then $G(w_1) = G(w_2)$ under $\mathcal{G}_{sta(n)}$. Moreover, if $G(w_1) = G(w_2)$ under $\mathcal{G}_{max(n)}$ and $span(w_1) \neq span(w_2)$, then $G(w_1) \neq G(w_2)$ under $\mathcal{G}_{sta(n)}$ from Lemma 2. Thus, $G(w_1) = G(w_2)$ under $\mathcal{G}_{sta(n)}$ if and only if $\mathcal{G}_{max(n)}$ and $span(w_1) = span(w_2)$. From Theorem 1, the set of the following (general) rules are sufficient to describe equivalence under $\mathcal{G}_{max(n)}$ when $v_1, v_2 \in \Sigma_n^*$.

1. $v_1\gamma_i\overline{\gamma_j}v_2 \leftrightarrow v_1\overline{\gamma_j}\gamma_i v_2$
2. $v_1\gamma_i\gamma_i v_2 \leftrightarrow v_1\gamma_i v_2$
3. $v_1\gamma_i\gamma_j v_2 \leftrightarrow v_1\gamma_j\gamma_i v_2$ for $|i - j| \geq 2$
4. $v_1\gamma_i\gamma_j\gamma_i v_2 \leftrightarrow v_1\gamma_i v_2$ for $|i - j| = 1$

Now, there exists a set P of a pair (w_1, w_2) of words where $G(w_1) = G(w_2)$ under $\mathcal{G}_{max(n)}$ and $\mathcal{G}_{sta(n)}$, a set H of a pair (w_3, w_4) of words where $G(w_3) = G(w_4)$ under $\mathcal{G}_{max(n)}$ and $G(w_3) \neq G(w_4)$ under $\mathcal{G}_{sta(n)}$, and the other set N of a pair (w_5, w_6) of words where $G(w_5) \neq G(w_6)$ under $\mathcal{G}_{max(n)}$ and $\mathcal{G}_{sta(n)}$. The sets P, H, N are disjoint and $P \cup H \cup N = \Sigma_n^* \times \Sigma_n^*$. For each pair in P (H), there exists the set of all sequences of rules in $R_{max(n)}$ that rewrites w_1 as w_2 (w_3 as w_4). Now, $R_{max(n)}$ can be partitioned into two sets R_{diff} and R_{same}, where all rules in R_{diff} have different span for two sides and all rules in R_{same} have the same span for two sides. Then, the following statements hold:

1. For a pair $(w_1, w_2) \in P$, there exists a sequence of rules that rewrites w_1 as w_2 only using rules in R_{same}: For a pair $(w_1, w_2) \in P$, there exists a word w_3 which is irreducible in $\mathcal{O}_{max(n)}$ and $(w_1, w_3), (w_2, w_3) \in P$. From Lemma 1, there exists a sequence of non-increasing rules that rewrite w_1 as w_3. We observe that the only rules that changes the size of the word are rule 2 and 4. Rule 2 does not change the span, and non-increasing rules in rule 4 do not increase the size of the span. Since $w_1 \sim w_3$ under $\mathcal{G}_{sta(n)}$, $span(w_1) = span(w_3)$. Thus, there exists a sequence of rules that rewrites w_1 as w_3 only using rules in R_{same}. The same statement holds for w_2, which yields a sequence of rules that rewrites w_1 as w_2 only using rules in R_{same}.
2. For a pair $(w_3, w_4) \in H$, all sequences of rules that rewrite w_3 as w_4 have a rule from R_{diff}.

Since $R_{same} \subseteq R_{max(n)}$, for a pair $(w_5, w_6) \in N$, w_5 cannot be rewritten as w_6 using R_{same}. Then, we can claim that a pair (w_1, w_2) is in P if and only if $w_1 \to_* w_2$ using R_{same}. Rules 1 to 3 from $R_{max(n)}$ have the same span for the both sides, and they are in R_{same}. For rule 4 from $R_{max(n)}$, the subset of the rules where both sides have the same span is rules 4 to 7 in $R_{sta(n)}$. Thus, $R_{sta(n)} = R_{same}$ and $G(w_1) = G(w_2)$ under $\mathcal{G}_{sta(n)}$ if and only if $w_1 \rightarrow_* w_2$ using $R_{sta(n)}$.

E Proof of Theorem 5

	ϵ	α_1	α_2	$\alpha_1\alpha_2$	$\alpha_2\alpha_1$	$\alpha_1\alpha_2\alpha_1$	$\alpha_2\alpha_1\alpha_2$
ϵ	ϵ	α_1	α_2	$\alpha_1\alpha_2$	$\alpha_2\alpha_1$	$\alpha_1\alpha_2\alpha_1$	$\alpha_2\alpha_1\alpha_2$
β_1	β_1	$\alpha_1\beta_1$	$\alpha_2\beta_1$	$\alpha_1\alpha_2\beta_1$	$\alpha_2\alpha_1\beta_1$	$\alpha_1\alpha_2\alpha_1\beta_1$	
β_2	β_2	$\alpha_1\beta_2$	$\alpha_2\beta_2$	$\alpha_1\alpha_2\beta_2$	$\alpha_2\alpha_1\beta_2$		$\alpha_2\alpha_1\alpha_2\beta_2$
$\beta_1\beta_2$	$\beta_1\beta_2$	$\alpha_1\beta_1\beta_2$	$\alpha_2\beta_1\beta_2$	$\alpha_1\alpha_2\beta_1\beta_2$	$\alpha_2\alpha_1\beta_1\beta_2$		
$\beta_2\beta_1$	$\beta_2\beta_1$	$\alpha_1\beta_2\beta_1$	$\alpha_2\beta_2\beta_1$	$\alpha_1\alpha_2\beta_2\beta_1$	$\alpha_2\alpha_1\beta_2\beta_1$		
$\beta_1\beta_2\beta_1$	$\beta_1\beta_2\beta_1$						
$\beta_2\beta_1\beta_2$	$\beta_2\beta_1\beta_2$						

Fig. 12. The set of representative words in $\mathcal{O}_{sta(3)}$. Gray headers represent representative words corresponding to elements in $\mathcal{J}_3$, and the thick box represents the set of representative words in $\mathcal{O}_{max(3)}$.

Fig. 12 enumerates representative words in $\mathcal{O}_{sta(3)}$. We observe that the set of representative words in $\mathcal{O}_{sta(n)}$ is a superset of the set of representative words in $\mathcal{O}_{max(n)}$. Graphically, we observe that $\mathcal{G}_{sta(n)}$ is a superset of the set of a pair of diagrams of $\mathcal{J}_n$, where we regard one as scaffolds and the other as staples. In particular, when a consecutive set of columns adjacent to the span is occupied with both real straight scaffold and staple, there also exists a structure with virtual straight staples in these columns as in Fig. 13.

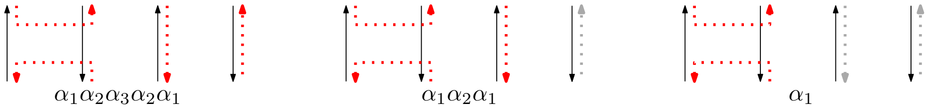

$\alpha_1\alpha_2\alpha_3\alpha_2\alpha_1$
$\alpha_1\alpha_2\alpha_1$
α_1

Fig. 13. Since the third and the fourth columns of $\alpha_1\alpha_2\alpha_3\alpha_2\alpha_1$ are occupied with both real straight scaffold and staple, we also have structures where the staple at the fourth column is virtual ($\alpha_1\alpha_2\alpha_1$) and staples at the third and the fourth columns are virtual (α_1).

We can classify graphical structures in $\mathcal{G}_{max(n)}$ by using a binary b of length n, where the ith digit has 1 if the ith column has both straight scaffold and staple, and 0 otherwise. The set of binaries of length n has bijection with the set T_n previously defined. Thus, let $D(p)$ be the number of equivalence classes of words whose graphical structures correspond to $p \in T_n$. It is straightforward that $D(0,1) = 1$ and $D(1,0) = 0$. To calculate $D(a_1, b_1, \ldots, a_k, b_k)$, columns that has 1's in the binary has only one case (straight scaffold and staple), so we need to multiply all $D(a_i, 0)$'s. For $D(a_1, 0)$, once we know all $D(p)$ where $p \in T_{a_1}$, we can calculate $D(a_1, 0)$ using the fact that $|\mathcal{G}_{max(a_1)}|^2 = \left(\frac{1}{a_1+1}\binom{2a_1}{a_1}\right)^2$ is equal to $\sum_{p \in T_{a_1}} D(p)$.

For each $D(p)$, we calculate the number of distinct graphical structures in $\mathcal{G}_{sta(n)}$. Suppose we have j consecutive 1's at the start or the end of the binary that corresponds to p. For such sequence, we can have $j+1$ distinct sets of virtual straight staples. When we have j consecutive 1's between two consecutive 0's, there are $\sum_{i=1}^{j} i + 1 = \frac{j(j+1)}{2} + 1$ distinct sets of virtual straight staples. The number of cases for each consecutive 1's should be multiplied, which results in $x(p)$ in the Theorem. The only exception for this calculation is $D(0, n)$ case, where we have real straight scaffolds and staples for all columns. The only word that corresponds to the structure is the empty word ϵ, and we need to subtract n from $D(0, n) \cdot x(0, n) = n + 1$, which results in the formula in the theorem. Fig. 14 shows how we count the number of cases for each $D(p)$.

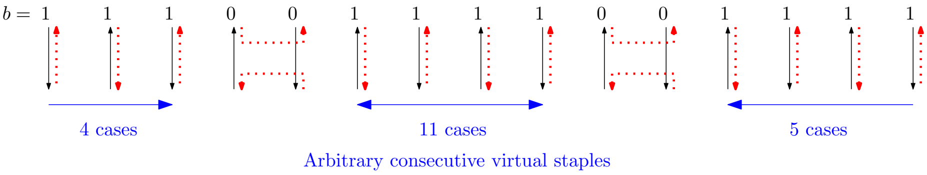

$b =$ 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1
4 cases
11 cases
5 cases
Arbitrary consecutive virtual staples

Fig. 14. A graphical structure corresponding to $D(0, 3, 2, 4, 2, 4)$. The binary that corresponds to the structure is stated on the structure. For consecutive 1's in the binary, we may have arbitrary consecutive virtual staples within.

To justify the counting of virtual staples cases, we claim the following statement: For a graphical structure in $\mathcal{G}_{sta(n)}$, let t_i be 1 if the ith column has real straight scaffold and straight staple, and 0 otherwise. Then, a set of maximal consecutive columns with real straight scaffolds and staples should be adjacent to the ith column where $t_i = 0$, as in Fig. 15 (a). In other words, there is no set of maximal consecutive columns with real straight scaffolds and staples where both ends are adjacent to straight scaffolds with straight virtual staples, as in Fig. 15 (b). We prove the statement by induction on the size of the word. It is straightforward that the statement holds for the generators. Assume that the statement holds for all $|w| = m$. For a word $w' = w\gamma_i$, we observe that a column in the graphical structure of w' can have a virtual straight staple only if it had a virtual straight staple in the graphical structure of w. Thus, if i or $i+1$ is in $span(w)$, the statement holds since the columns that have virtual straight staples do not change. If i and $i+1$ are not in $span(w)$, we insert the unit of the generator in a set of consecutive columns with straight scaffolds and virtual staples. In that case, concatenation does not create columns that have straight scaffolds and real staples, and the statement holds.

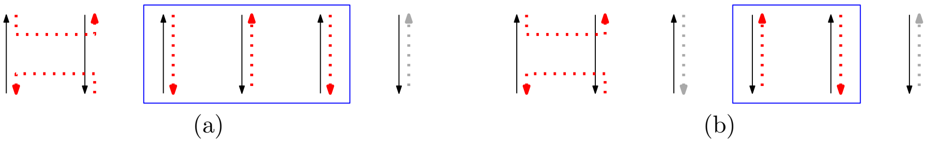


Fig. 15. The set of maximal consecutive columns with real straight scaffolds and staples is represented by a blue box. (a) The set is adjacent to the ith column with $t_i = 0$ on the left. (b) Both ends are adjacent to straight scaffolds with straight virtual staples, which is impossible.

F Proof of Theorem 6

There exists a representative word $w_a w_b \in \mathcal{O}_{sta(n)}$ in a inter-commutation-free form where $|w_a| = |w_b| = \left\lfloor \frac{n^2}{4} \right\rfloor$, and $u(n) \geq \left\lfloor \frac{n^2}{4} \right\rfloor$. Aside from $w_a w_b$, we may have representative words in $\mathcal{O}_{sta(n)}$ that exploit rules 4 to 7, not satisfying the condition that $\gamma_i\gamma_j\gamma_i$ is reduced to γ_i when $|i-j| = 1$. Namely, we may rewrite γ_i in a word w as $\gamma_i\gamma_j\gamma_i$ and have a distinct word if staples in the resulting word occupy at least one new column.

Given an irreducible word v of size $l(v)$, let $s(v)$ be the size of the span of v. Then, $\frac{s(v)}{2} \leq l(v) \leq u(s(v)-1)$ holds when $v \neq \epsilon$. Now, suppose for a word $v = v_a v_b$ in a inter-commutation-free form, we want to continuously rewrite γ_i as $\gamma_i\gamma_j\gamma_i$ as far as possible while making the resulting words distinct. Let $v_a = \alpha_{i_1} \cdots \alpha_{i_p}$ and $v_b = \beta_{j_1} \cdots \beta_{j_q}$. If $v_a = \epsilon$, for a resulting word v', there exists a longer word $\alpha_{j_1} \cdots \alpha_{j_q} v'$ which is distinct. Thus, without loss of generality, we may assume that $v_a, v_b \neq \epsilon$. Now, for the word v, $l(v) = l(v_a) + l(v_b)$ and $\max(s(v_a), s(v_b)) \leq s(v) \leq s(v_a) + s(v_b)$. Each rewriting of γ_i to $\gamma_i\gamma_j\gamma_i$ increases s by 1 and l by 2, and such rewriting becomes impossible once s becomes $n+1$. Without loss of generality, we assume that $s(v_a) \geq s(v_b)$. Then, we may have at most $n+1-s(v_a)$ number of rewriting steps, which results in a word of size $l(v_a) + l(v_b) + 2(n+1-s(v_a)) = 2n+2-2s(v_a) + l(v_a) + l(v_b) \leq 2n+2-2s(v_a)+2u(s(v_a)-1)$. Since $u(n) \geq \left\lfloor \frac{n^2}{4} \right\rfloor$, $2n+2-2s(v_a)+2u(s(v_a)-1)$ increases as $s(v_a)$ increases. When $s(v_a) = n+1$, $u(n) \leq 2n+2-2(n+1)+2u(n)$, which results in $u(n) \geq 0$, which is trivial. For $s(v_a) = n$, $u(n) \leq 2n + 2 - 2n + 2u(n-1) = 2u(n-1) + 2$. From $u(1) = 0$, we have the upper bound $u(n) = 2^n - 2$.