Green's Relations
in Finite Transformation Semigroups

Lukas Fleischer$^{(\boxtimes)}$ and Manfred Kufleitner

FMI, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany
{fleischer,kufleitner}@fmi.uni-stuttgart.de

Abstract. We consider the complexity of Green's relations when the semigroup is given by transformations on a finite set. Green's relations can be defined by reachability in the (right/left/two-sided) Cayley graph. The equivalence classes then correspond to the strongly connected components. It is not difficult to show that, in the worst case, the number of equivalence classes is in the same order of magnitude as the number of elements. Another important parameter is the maximal length of a chain of components. Our main contribution is an exponential lower bound for this parameter. There is a simple construction for an arbitrary set of generators. However, the proof for constant alphabet is rather involved. Our results also apply to automata and their syntactic semigroups.

1 Introduction

Let Q be a finite set with n elements. There are n^n mappings from Q to Q. Such mappings are called *transformations* and the elements of Q are called *states*. The composition of transformations defines an associative operation. If Σ is some arbitrary subset of transformations, we can consider the *transformation semigroup* S generated by Σ; this is the closure of Σ under composition.[1] The set of all transformations on Q is called the *full transformation semigroup* on Q. One can view (Q, Σ) as a description of S. Since every element s of a semigroup S defines a transformation $x \mapsto x \cdot s$ on $S^1 = S \cup \{1\}$, every semigroup S admits such a description (S^1, S); here, 1 either denotes the neutral element of S or, if S does not have a neutral element, we add 1 as a new neutral element. Essentially, the description (S^1, S) is nothing but the multiplication table for S. On the other hand, there are cases where a description as a transformation semigroup is much more succinct than the multiplication table. For instance, the full transformation semigroup on Q can be generated by a set Σ with three elements [7]. In addition to the size of S, it would be interesting to know which other properties could be derived from the number n of states.

Green's relations are an important tool for analyzing the structure of a semigroup S. They are defined as follows:

---

[1] When introducing transformation semigroups in terms of actions, then this is the framework of *faithful* actions.

$$s \leqslant_{\mathcal{R}} t \text{ if } sS^1 \subseteq tS^1, \quad s \leqslant_{\mathcal{L}} t \text{ if } S^1 s \subseteq S^1 t, \quad s \leqslant_{\mathcal{J}} t \text{ if } S^1 sS^1 \subseteq S^1 tS^1.$$

We write $s \, \mathcal{R} \, t$ if both $s \leqslant_{\mathcal{R}} t$ and $s \leqslant_{\mathcal{R}} t$; and we set $s <_{\mathcal{R}} t$ if $s \leqslant_{\mathcal{R}} t$ but not $s \, \mathcal{R} \, t$. The relations $\mathcal{L}, <_{\mathcal{L}}, \mathcal{J}$ and $<_{\mathcal{J}}$ are defined analogously. The relations $\mathcal{R}$, $\mathcal{L}$, and $\mathcal{J}$ form equivalence relations. The equivalence classes corresponding to these relations are called $\mathcal{R}$-*classes* (resp. $\mathcal{L}$-*classes*, $\mathcal{J}$-*classes*) of S. Instead of ideals, one could alternatively also use reachability in the right (resp. left, two-sided) Cayley graph of S for defining $\leqslant_{\mathcal{R}}$ (resp. $\leqslant_{\mathcal{L}}, \leqslant_{\mathcal{J}}$). We note that $s <_{\mathcal{R}} t$ implies $s <_{\mathcal{J}} t$ and, symmetrically, $s <_{\mathcal{L}} t$ implies $s <_{\mathcal{J}} t$. The complexity of deciding Green's relations for transformation semigroups was recently shown to be PSPACE-complete [1]. When considering a transformation semigroup on n states, one of our first results shows that the maximal number of $\mathcal{J}$-classes is in $n^{\Theta(n)}$. In particular, the number of equivalence classes is in the same order of magnitude as the size of the transformation semigroup. Since every $\mathcal{J}$-class contains at least one $\mathcal{R}$- and one $\mathcal{L}$-class, the same bound holds for $\mathcal{R}$ and $\mathcal{L}$.

   Another important parameter is the maximal length ℓ such that there are elements $s_1, \ldots, s_\ell$ with $s_1 >_{\mathcal{R}} \cdots >_{\mathcal{R}} s_\ell$, called the $\mathcal{R}$-*height*. Similarly, we are interested in the $\mathcal{L}$- and $\mathcal{J}$-height. Many semigroup constructions such as the Rhodes expansion and variants thereof rely on this parameter; see e.g. [2,3,6]. We show that the maximal $\mathcal{R}$-height is in $2^{\Theta(n)}$; for the maximal $\mathcal{L}$-height and $\mathcal{J}$-height we only have $2^{\Omega(n)}$ as a lower bound. Proving the lower bounds for a fixed number of generators is much more involved than for arbitrarily many generators. The exponential lower bounds are quite unexpected in the following sense: If the transformation semigroup is small, then the number of equivalence classes (and hence, the lengths of chains) cannot be big. On the other hand, the transformation semigroup is maximal if it is full. And an equivalence class in the full transformation semigroup only depends on the number of states in the image; this is because we can apply arbitrary permutations. In particular, the number of equivalence classes in these two extreme cases is small.

   There is a tight connection between deterministic automata and transformation semigroups. Roughly speaking, a transformation semigroup is an automaton without initial and finial states. The main difference is that for automata, one usually is interested in the syntactic semigroup rather than the transformation semigroup; the syntactic semigroup is the transformation semigroup of the minimal automaton. We show that the above bounds on the number of equivalence classes and heights also apply to syntactic semigroups.

Theorem 1. *For each $n \in \mathbb{N}$, there exists a minimal automaton $\mathcal{A}_n$ with n states over an alphabet of size 5 such that the number of $\mathcal{J}$-classes (resp. $\mathcal{R}$-classes, $\mathcal{L}$-classes) of the transformation semigroup $T(\mathcal{A}_n)$ is at least $(n-4)^{n-4}$.*

Theorem 2. *There exists a sequence of minimal automata $(\mathcal{A}_n)_{n \in \mathbb{N}}$ over a fixed alphabet such that $\mathcal{A}_n$ has n states and the $\mathcal{R}$-height (resp. $\mathcal{L}$-height, $\mathcal{J}$-height) of the transformation semigroup $T(\mathcal{A}_n)$ is in $\Omega(2^n / n^{9.5})$.*

2   Preliminaries

A *semigroup* is a set S equipped with an associative operation $\cdot\colon S \times S \to S$. A *subsemigroup* of S is a subset T such that $s_1 s_2 \in T$ for all $s_1, s_2 \in T$. It is called *completely isolated* if the converse implication holds, i.e., $s_1 s_2 \in T$ implies $s_1 \in T$ and $s_2 \in T$ for all $s_1, s_2 \in S$. The *opposite* semigroup of S is obtained by replacing the operation with its left-right dual $\circ\colon S \times S \to S$ defined by $x \circ y = y \cdot x$.

In general, Green's relations in a subsemigroup T of S do not coincide with the corresponding relations in S. However, if T is a completely isolated subsemigroup, the following property holds:

Proposition 3. *Let S be a semigroup and let T be a completely isolated subsemigroup of S. Let $\mathcal{K}$ be one of the relations $\leqslant_\mathcal{R}$, $\leqslant_\mathcal{L}$, $\leqslant_\mathcal{J}$, $\mathcal{R}$, $\mathcal{L}$ or $\mathcal{J}$. Then, for all $x, y \in T$, we have $x \,\mathcal{K}\, y$ in S if and only if $x \,\mathcal{K}\, y$ in T.*

Proof. We will only prove the statement for the preorder $\leqslant_\mathcal{R}$. For the implication from right to left, we have $xS^1 \subseteq xT^1 S^1 \subseteq yT^1 S^1 \subseteq yS^1$. For the converse implication, suppose that $xS^1 \subseteq yS^1$, i.e., there exists some $z \in S^1$ such that $yz = x$. Since T is completely isolated, we have $z \in T^1$, which yields $zT^1 \subseteq T^1$ and thus, $xT^1 = yzT^1 \subseteq yT^1$. $\qquad\qquad\square$

An $\mathcal{R}$-*chain* is a sequence $(s_1, \dots, s_\ell)$ of elements of S such that $s_{i+1} <_\mathcal{R} s_i$ for all $i \in \{1, \dots, \ell - 1\}$; ℓ is called the *length* of the $\mathcal{R}$-chain. The maximal length of an $\mathcal{R}$-chain of S is called the $\mathcal{R}$-*height* of S. The notions $\mathcal{L}$-chains, $\mathcal{J}$-chains, $\mathcal{L}$-height and $\mathcal{J}$-height are defined analogously.

A *partial transformation* on a set Q is a partial function $f\colon Q \to Q$. If the domain of f is all of Q, i.e., if f is a total function, f is called a *transformation*. A partial transformation $f\colon Q \to Q$ is called *injective* if $f(p) \neq f(q)$ whenever $p \neq q$ and both $f(p)$ and $f(q)$ are defined. The elements of Q are often called *states*. In the following, we use the notation $q \cdot f$ instead of $f(q)$ to denote the image of an element $q \in Q$ under f. For $R \subseteq Q$ let $R \cdot f = \{q \cdot f \mid q \in R\}$. Note that for all subsets $R \subseteq Q$ and all partial transformations $f\colon Q \to Q$, the inequality $|R \cdot f| \leqslant |R|$ holds; we will implicitly use this property throughout the paper. The *composition* fg of two transformations $f\colon Q \to Q$ and $g\colon Q \to Q$ is defined by $q \cdot fg = (q \cdot f) \cdot g$. The composition is associative.

The set of all partial transformations (resp. transformations) on a fixed set Q forms a semigroup with composition as the binary operation. It is called the *full partial transformation semigroup* (resp. *full transformation semigroup*) on Q. Subsemigroups of full (partial) transformation semigroups are called *(partial) transformation semigroups* and are often specified in terms of generators. Partial transformation semigroups and transformation semigroups are strongly related. On one side, every transformation semigroup also is a partial transformation semigroup. In the other direction a slightly weaker statement holds:

Proposition 4. *Let P be a partial transformation semigroup on n states. Then there exists a transformation semigroup on $n+1$ states which is isomorphic to P.*

A partial transformation semigroup is called *injective* if it is generated by a set of injective partial transformations. An important property of injective partial transformation semigroups is that they have a left-right dual:

Proposition 5. *The opposite semigroup of an injective partial transformation semigroup is a partial transformation semigroup.*

Transformation semigroups naturally arise when considering deterministic finite automata. Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be a deterministic finite automaton. Then, each letter $a \in \Sigma$ can be interpreted as a transformation $a \colon Q \to Q$ where $q \cdot a = \delta(q, a)$. The transformation semigroup on Q generated by all letters in Σ is denoted by $T(\mathcal{A})$ and it is called the *transition semigroup* of $\mathcal{A}$. Conversely, given a transformation semigroup T on a finite set Q and a finite set of generators Σ, for each $q_0 \in Q$ and $F \subseteq Q$, one can define a deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where $\delta \colon Q \times \Sigma \to Q$ is defined as $\delta(q, a) = q \cdot a$.

A well-known approach for translating bounds on the size of a transformation semigroup to syntactic monoids is to *make* an automaton minimal. This can be done by introducing a new generator c with $q_i \cdot c = q_{i+1}$ for $Q = \{q_1, \ldots, q_n\}$ and $q_{n+1} = q_1$; moreover, one chooses some arbitrary state to be both initial and final. We adapt this construction to also work with Green's relations.

Proposition 6. *Let T be a transformation semigroup on n states, generated by a finite set Σ. Then there exists a minimal $(n+1)$-state deterministic finite automaton $\mathcal{A}$ over an alphabet of size $|\Sigma|+1$ such that T is a completely isolated subsemigroup of $T(\mathcal{A})$.*

Proof. Let T be a transformation semigroup on a set of states $Q = \{q_1, \ldots, q_n\}$, generated by Σ. Let $\mathcal{A} = (Q \cup \{q_0\}, \Sigma \cup \{c\}, \delta, q_0, \{q_n\})$ be the automaton defined by $\delta(q_0, a) = q_0$ and $\delta(q_i, a) = q_i \cdot a$ for $i \geqslant 1$ and all $a \in \Sigma$. The transitions for the letter c are defined by $\delta(q_i, c) = q_{i+1}$ for $i < n$ and $\delta(q_n, c) = q_1$. This automaton is minimal: for two different states $q_i, q_j \in Q \cup \{q_0\}$ with $i > j$, we have $\delta(q_i, c^{n-i}) = q_n$ but $\delta(q_j, c^{n-i}) \neq q_n$.

By construction, T is a subsemigroup of $T(\mathcal{A})$. To see that T is completely isolated within $T(\mathcal{A})$, note that we have $\delta(q_0, u) = q_0$ if and only if $u \in \Sigma^*$. $\quad\square$

3   Bounds for the Number of Classes

Let $\mathcal{K}$ be any of the relations $\mathcal{R}$, $\mathcal{L}$ or $\mathcal{J}$. The naïve upper bound for the number of $\mathcal{K}$-classes of a transformation semigroup T on n states is given by the size of T itself. Since there are n^n different functions from Q to Q, the semigroup T contains at most n^n elements. It is well known that this bound is tight even for a constant number of generators, since for each $n \geqslant 1$ there exists a transformation semigroup of size n^n generated by a set Σ with three elements; see e.g. [7].

As each $\mathcal{R}$-class (resp. $\mathcal{L}$-class, $\mathcal{J}$-class) consists of at least one element, the number of such classes is also bounded by n^n. We now show that this upper bound is tight up to a constant factor.

Proposition 7. *Let T be a transformation semigroup on n states, generated by a finite set Σ. Then there exists a transformation semigroup on $n + 3$ states which is generated by $|\Sigma| + 1$ elements and has at least $|T|$ different $\mathcal{J}$-classes.*

Proof. Let T be a semigroup of transformations on a set of states Q, generated by a finite set Σ, and let q_0 be an arbitrary element from Q. Let q_1, q_2, q_3 be new states not in Q and let c be a new generator not in Σ. Let U be the transformation semigroup on $Q \cup \{q_1, q_2, q_3\}$ obtained by extending the transformations of T as follows: for each $a \in \Sigma$ and $q \in Q$, let $q \cdot c = q$, $q_1 \cdot a = q_3 \cdot a = q_3 \cdot c = q_0$, $q_1 \cdot c = q_2 \cdot a = q_2$, and $q_2 \cdot c = q_3$.

Let $u, v \in \Sigma^*$ be different elements of T. Then cuc and cvc are different in U. We claim that $cuc \not\leqslant_{\mathcal{J}} cvc$ in U. For the sake of contradiction, suppose that there exist $x, y \in (\Sigma \cup \{c\})^*$ such that $cuc = xcvcy$ in U. Clearly, $q_1 \cdot cuc = q_3 \notin Q$. Moreover, at least one of the words x or y must be non-empty and therefore $q_1 \cdot xcucy \in Q$. This shows that $cuc \neq xcvcy$, as desired. □

Combining the result with statements from the previous section, we obtain a lower bound for the number of $\mathcal{J}$-classes of the transition semigroup of an automaton.

Proof (Theorem 1). As we mentioned before, it is well known that there exists a 3-generator transformation semigroup on n states of size n^n. If we first apply Proposition 7 and then Proposition 6 to T, we obtain the claim by Proposition 3. The statement extends to $\mathcal{R}$-classes (resp. $\mathcal{L}$-classes) because each $\mathcal{J}$-class contains at least one $\mathcal{R}$-class (resp. $\mathcal{L}$-class). □

4   Bounds for the Length of Chains

Let $\mathcal{K}$ be any of the relations $\mathcal{R}$, $\mathcal{L}$ or $\mathcal{J}$. As with the number of $\mathcal{K}$-classes, the naïve upper bound for the length of $\mathcal{K}$-chains is given by the maximal size n^n of the transformation semigroup on n states. In this section, we improve this upper bound for $\mathcal{R}$-chains and later give a lower bound that matches up to a polynomial gap.

Lemma 8. *Let P be a partial transformation semigroup on a finite set Q of cardinality n. Let $x, y \in P$ such that $Q \cdot x = Q \cdot xy$. Then $x \mathrel{\mathcal{R}} xy$.*

Proof. Let $\omega = n!$ and let $z = y^{\omega - 1}$. It suffices to show that $xyz = x$ in P, i.e., for all $q \in Q$, we have $q \cdot x = q \cdot xyz$. By assumption, the restriction of y to the set $Q \cdot x$ is bijective. Thus, the mapping y^ω acts as identity on $Q \cdot x$. This yields $q \cdot xyz = q \cdot xy^\omega = (q \cdot x) \cdot y^\omega = q \cdot x$. □

Proposition 9. *Let P be a partial transformation semigroup on n states. Then the $\mathcal{R}$-height of P is at most 2^n.*

Proof. Let P be a partial transformation semigroup on a set of states Q with $|Q| = n$. Let $(u_1, u_2, \ldots, u_\ell)$ be an $\mathcal{R}$-chain of P. We show that all sets $Q \cdot u_i$ must be pairwise distinct which yields the desired bound. Suppose that $Q \cdot u_i = Q \cdot u_j$ for $1 \leqslant i < j \leqslant \ell$. Since $u_j <_{\mathcal{R}} u_i$, there exists $v \in P$ with $u_i v = u_j$. Lemma 8 yields $u_j \mathrel{\mathcal{R}} u_i$ which implies $u_{i+1} \mathrel{\mathcal{R}} u_i$, a contradiction. □

4.1 Token Computations in Transformation Semigroups

In this subsection, we introduce the building blocks for the lower bound on the height. A *token machine* is a pair (C, I) where C is a finite set and I is a set of partial transformations on C. The elements of the set C are called *cells*, subsets of C are called *configurations* and the generators I are called *instructions*.

A *program* is a finite word over the alphabet I and a *computation* is a sequence

$$R_0 \xrightarrow{\iota_1} R_1 \xrightarrow{\iota_2} R_2 \cdots \xrightarrow{\iota_\ell} R_\ell$$

where all $R_i \subseteq C$ have the same cardinality and $R_{i-1} \cdot \iota_i = R_i$. The configuration R_0 is called *initial configuration* and R_ℓ is called the *final configuration* of the computation. The program $\iota_1 \iota_2 \cdots \iota_\ell$ is the *label* of the computation and ℓ is its *length*. It is *progressing* if all configurations appearing in the computation are pairwise distinct and for each $i \in \{1, \dots, \ell\}$ and each $\iota \in I \setminus \{\iota_i\}$, we have $|R_{i-1} \cdot \iota| < |R_i|$. It is *maximal* if $|R_\ell \cdot \iota| < |R_\ell|$ for all $\iota \in I$.

A language over programs $L \subseteq I^*$ is called *deterministic* on a configuration $R \subseteq C$ if $|R \cdot u_1| = |R| = |R \cdot u_2|$ implies $u_1 = u_2$ for all $u_1, u_2 \in L$.

The focal idea of token machines is captured in the following proposition which states that computations in token machines naturally yield lower bounds for the length of $\mathcal{R}$-chains.

Proposition 10. *Let (C, I) be a token machine and let P be the partial transformation semigroup on C generated by I. If there exists a maximal progressing computation of length ℓ, then the $\mathcal{R}$-height of P is at least ℓ.*

Proof. Let $R_0 \xrightarrow{\iota_1} R_1 \xrightarrow{\iota_2} R_2 \cdots \xrightarrow{\iota_\ell} R_\ell$ be a maximal progressing computation. For each $i \in \{1, \dots, \ell\}$, we let $u_i = \iota_1 \iota_2 \cdots \iota_i$. It remains to show that $(u_1, \dots, u_\ell)$ is an $\mathcal{R}$-chain. By definition, we immediately obtain $u_{i+1} \leqslant_\mathcal{R} u_i$. Assume, for the sake of contradiction, that $u_i \leqslant_\mathcal{R} u_{i+1}$ for some $i \in \{1, \dots, \ell - 1\}$, i.e., there exists $v \in I^*$ with $u_i = u_{i+1} v$. Without loss of generality, we may assume that i is maximal with this property. If $|v| = 0$, then $R_i = R_0 \cdot u_i = R_0 \cdot u_{i+1} = R_{i+1}$, contradicting the premise of progression. Thus, $|v| \geqslant 1$ and since the computation is progressing and maximal, we have $i < \ell - 1$ and $v = \iota_{i+2} w$ for some $w \in I^*$. This yields $u_{i+2} w \iota_{i+1} = u_{i+1} \iota_{i+2} w \iota_{i+1} = u_{i+1} v \iota_{i+1} = u_i \iota_{i+1} = u_{i+1}$, contradicting the maximality of i. $\qquad\square$

4.2 Lower Bounds over a Growing Instruction Set

Before describing the technical ingredients required in our main result, we prove a slightly weaker statement. In contrast to the result presented later, it relies on an alphabet that grows exponentially with the number of elements.

Theorem 11. *For all even $n \in \mathbb{N}$, there exists a token machine with n cells which admits a maximal progressing computation of length at least $\binom{n}{n/2} - 1$.*

Proof. Let $C = \{1, 2, \ldots, n\}$. Let $\ell = \binom{n}{n/2} - 1$ and let $\{R_0, R_1, \ldots, R_\ell\}$ be the set of $n/2$-element subsets of C. For each $i \in \{1, \ldots, \ell\}$, let $\iota_i \colon R_{i-1} \to R_i$ be a bijection. Note that in the context of the present proof, it does not matter which of the $(n/2)!$ bijections is chosen; for example, one can always choose the unique bijection ι_i such that $\iota_i(j) < \iota_i(k)$ if and only if $j < k$. Each ι_i can be viewed as a partial transformation on C which is undefined for all $c \in C \backslash R_{i-1}$. We now show that in the token machine (C, I) with $I = \{\iota_i \mid 1 \leqslant i \leqslant \ell\}$, the sequence

$$R_0 \xrightarrow{\iota_1} R_1 \xrightarrow{\iota_2} R_2 \cdots \xrightarrow{\iota_\ell} R_\ell$$

is a maximal progressing computation. It is a valid computation by the definition of the instructions ι_i. Consider $i \in \{0, \ldots, \ell\}$ and $j \in \{1, 2, \ldots, \ell\} \backslash \{i+1\}$. Since $R_{j-1} \neq R_i$, the instruction ι_j is undefined on at least one element of R_i and thus, $|R_i \cdot \iota_j| < |R_i|$. This shows that the computation is both progressing and maximal. □

The theorem has a series of interesting consequences which will be outlined in Sect. 4.4, after proving an improved variant of the theorem with fixed alphabet.

4.3   Tapes and Binary Counters

A *sub-machine* of a token machine (C, I) is a subset $S \subseteq C$ such that for each configuration R and for each instruction $\iota \in I$ with $|R \cdot \iota| = |R|$, we also have $|(R \cap S) \cdot \iota| = |R \cap S|$. In other words, each computation stays a computation when restricted to S. The *union* of two token machines (C, I) and (C', I') with $C \cap C' = \emptyset$ is the token machine $(C \cup C', I \cup I')$ where the instructions in $I \backslash I'$ are extended to act as identity on C' and the instructions in $I' \backslash I$ are extended to act as identity on C. The cells C and C' of the original machines are sub-machines of the union.

An *n-bit tape* T is a token machine (C, I) with n cells and an arbitrary (but fixed) order $(c_0, c_1, \ldots, c_{n-1})$. One can interpret configurations $R \subseteq C$ as bit strings $b_{n-1} b_{n-2} \cdots b_0$ where $b_i = 1$ if and only if $c_i \in R$ and $b_i = 0$ otherwise, and think of T as a ring buffer with a read/write head at position 0. An instruction ι_{rotl}^T can be used to move the tape head to the right (or, actually, retain the head position but left-rotate the buffer). For each $i \in \{0, \ldots, n-2\}$, we let $c_i \cdot \iota_{\mathsf{rotl}}^T = c_{i+1}$ and $c_{n-1} \cdot \iota_{\mathsf{rotl}}^T = c_0$. The instruction ι_{rotr}^T is defined analogously and moves in the opposite direction. An instruction $\iota_{=0}^T$ can be used to check whether the head is scanning a zero and halt the program otherwise. It is undefined on c_0 and defined as the identity on $\{c_1, \ldots, c_{n-1}\}$. Conversely, the ι_{sync}^T instruction is defined as the identity on c_0 and undefined on every other cell. An instruction ι_{mvl}^T maps c_0 to c_1, acts as the identity on $\{c_2, c_3, \ldots, c_{n-1}\}$ and is undefined on c_1. Analogously, ι_{mvr}^T maps c_0 to c_{n-1}, acts as the identity on $\{c_1, c_2, \ldots, c_{n-2}\}$ and is undefined on c_{n-1}. The *value* of T under a configuration R is $\sum_{c_i \in R} 2^i$.

An *n-bit binary counter* N is constructed as follows. Three new n-bit tapes S, T and $\overline{T}$ are introduced. Their cells are $(d_0, d_1, \ldots, d_{n-1})$, $(c_0, c_1, \ldots, c_{n-1})$ and $(\overline{c}_0, \overline{c}_1, \ldots, \overline{c}_{n-1})$, respectively. Then, the union of S, T and $\overline{T}$ is constructed and the following instructions are added:

$$- \iota_{\mathsf{rotl}}^{N} = \iota_{\mathsf{rotl}}^{T} \iota_{\mathsf{rotl}}^{\overline{T}} \iota_{\mathsf{rotl}}^{S}, \qquad - \iota_{=0}^{N} = \iota_{=0}^{T}, \qquad - \iota_{\mathsf{sync}}^{N} = \iota_{\mathsf{sync}}^{S},$$

- $\iota_{\mathsf{rotr}}^{N} = \iota_{\mathsf{rotr}}^{T} \iota_{\mathsf{rotr}}^{\overline{T}} \iota_{\mathsf{rotr}}^{S},$ $- \iota_{=1}^{N} = \iota_{=0}^{\overline{T}},$ $- \iota_{\mathsf{off}}^{N} = \iota_{=0}^{S},$
- ι_{inc}^{N} with $\overline{c}_0 \cdot \iota_{\mathsf{inc}}^{N} = c_0$ and $c_0 \cdot \iota_{\mathsf{inc}}^{N}$ undefined and $c \cdot \iota_{\mathsf{inc}}^{N} = c$ for all $c \notin \{c_0, \overline{c}_0\}$,
- ι_{dec}^{N} with $c_0 \cdot \iota_{\mathsf{dec}}^{N} = \overline{c}_0$ and $\overline{c}_0 \cdot \iota_{\mathsf{dec}}^{N}$ undefined and $c \cdot \iota_{\mathsf{dec}}^{N} = c$ for all $c \notin \{c_0, \overline{c}_0\}$.

Following this, the original instructions of S, T and $\overline{T}$ are removed from I. Thus, a binary counter provides exactly eight instructions. A configuration R of N is *valid* if $|R \cap S| = 1$ and for each $i \in \{0, \dots, n-1\}$, we have $c_i \in R$ if and only if $\overline{c}_i \notin R$.

Lemma 12. *Let R be a valid configuration of a binary counter N and let $u \in I^*$ such that $|R \cdot u| = |R|$. Then $R \cdot u$ is a valid configuration of N.*

Proof. By induction on the length of u, it suffices to prove that the action of instructions on R preserves validity.

The instructions $\iota_{\mathsf{rotl}}^{N}$ and $\iota_{\mathsf{rotr}}^{N}$ cyclically rotate the tapes T, $\overline{T}$ and S. Thus, if R is valid, then $R \cdot \iota_{\mathsf{rotl}}^{N}$ and $R \cdot \iota_{\mathsf{rotr}}^{N}$ are valid as well.

For each $\iota \in \left\{ \iota_{=0}^{N}, \iota_{=1}^{N}, \iota_{\mathsf{sync}}^{N}, \iota_{\mathsf{off}}^{N} \right\}$, we have either $|R \cdot \iota| < |R|$ or $R \cdot \iota = R$. If $\left| R \cdot \iota_{\mathsf{inc}}^{N} \right| = |R|$, then R does not contain c_0. If, moreover, R is a valid configuration, it contains $\overline{c}_0$. But then, $R \cdot \iota_{\mathsf{inc}}^{N}$ contains c_0 and does not contain $\overline{c}_0$. It coincides with R on all other cells. Thus, $R \cdot \iota_{\mathsf{inc}}^{N}$ is valid as well. By a symmetric argument, the instruction ι_{dec}^{N} preserves validity. $\square$

We now define three regular languages

$$L_{\mathsf{reset}}^{N} = \iota_{\mathsf{sync}}^{N} ((\iota_{=0}^{N} \mid \iota_{\mathsf{dec}}^{N}) \iota_{\mathsf{rotr}}^{N} \iota_{\mathsf{off}}^{N})^* (\iota_{=0}^{N} \mid \iota_{\mathsf{dec}}^{N}) \iota_{\mathsf{rotr}}^{N} \iota_{\mathsf{sync}}^{N},$$

$$L_{\mathsf{inc}}^{N} = \iota_{\mathsf{sync}}^{N} (\iota_{\mathsf{dec}}^{N} \iota_{\mathsf{rotr}}^{N} \iota_{\mathsf{off}}^{N})^* \iota_{\mathsf{inc}}^{N} (\iota_{\mathsf{off}}^{N} \iota_{\mathsf{rotr}}^{N})^* \iota_{\mathsf{sync}}^{N} \text{ and}$$

$$L_{\mathsf{dec}}^{N} = \iota_{\mathsf{sync}}^{N} (\iota_{\mathsf{inc}}^{N} \iota_{\mathsf{rotr}}^{N} \iota_{\mathsf{off}}^{N})^* \iota_{\mathsf{dec}}^{N} (\iota_{\mathsf{off}}^{N} \iota_{\mathsf{rotr}}^{N})^* \iota_{\mathsf{sync}}^{N}.$$

Lemma 13. *The languages L_{reset}^{N}, L_{inc}^{N} and L_{dec}^{N} are deterministic on all valid configurations.*

Proof. Suppose there are two different words $u_1, u_2 \in L_{\mathsf{reset}}^{N}$ and a valid configuration R such that $|R \cdot u_1| = |R|$. Since L_{reset}^{N} is prefix-free, there exist a unique word $p \in I^*$ and different instructions $\iota_1, \iota_2 \in I$ such that $u_1 \in p\iota_1 I^*$ and $u_2 \in p\iota_2 I^*$. A careful analysis of the structure of the regular expression for L_{reset}^{N} shows that either $\{\iota_1, \iota_2\} = \left\{\iota_{=0}^{N}, \iota_{\mathsf{dec}}^{N}\right\}$ or $\{\iota_1, \iota_2\} = \left\{\iota_{\mathsf{off}}^{N}, \iota_{\mathsf{sync}}^{N}\right\}$.

In the first case, we may assume without loss of generality that $\iota_1 = \iota_{=0}^{N}$ and $\iota_2 = \iota_{\mathsf{dec}}^{N}$. From $|R \cdot p\iota_1| = |R \cdot p|$, we deduce $c_0 \notin R \cdot p$ because $\iota_{=0}^{N}$ is undefined on c_0. This implies $\overline{c}_0 \in R \cdot p$ since $R \cdot p$ is a valid configuration by Lemma 12. Since ι_{dec}^{N} is undefined on $\overline{c}_0$, it follows that $|R \cdot u_2| \leqslant |R \cdot p\iota_2| < |R \cdot p| \leqslant |R|$.

In the second case, we may assume that $\iota_1 = \iota_{\mathsf{off}}^{N}$ and $\iota_2 = \iota_{\mathsf{sync}}^{N}$. Since $|R \cdot p\iota_1| = |R \cdot p|$ and since ι_{off}^{N} is undefined on d_0, we have $d_0 \notin R \cdot p$. This implies $d_i \in R \cdot p$ for some $i \in \{1, \dots, n-1\}$ because $R \cdot p$ is valid by Lemma 12. The instruction $\iota_{\mathsf{sync}}^{N}$ is undefined on $\{d_1, d_2, \dots, d_{n-1}\}$ which yields $|R \cdot u_2| < |R|$, as above.

The proofs for L_{inc}^{N} and L_{dec}^{N} follow by a similar reasoning. $\square$

Let R be a configuration of N. We say that the counter is *synchronized* under R if $d_0 \in R$. The *value* of N under R is the value of T under $R \cap \{c_0, c_1, \ldots, c_{n-1}\}$.

In addition to the eight counter instructions defined above, for any fixed constant $k \in \{0, \ldots, 2^n - 1\}$ one can define an instruction $\iota_{\mathsf{val}=k}^N$ which asserts that the value of the counter equals k as follows. For each $i \in \{0, \ldots, n-1\}$ with $k \bmod 2^{i+1} \geqslant 2^i$, we let $c_i \cdot \iota_{\mathsf{val}=k}^N = c_i$ and let $\bar{c}_i \cdot \iota_{\mathsf{val}=k}^N$ be undefined. Symmetrically, we let $\bar{c}_i \cdot \iota_{\mathsf{val}=k}^N = \bar{c}_i$ and $c_i \cdot \iota_{\mathsf{val}=k}^N$ undefined if $k \bmod 2^{i+1} < 2^i$.

Lemma 14. *Let R be a valid configuration and let $u \in L_{\mathsf{reset}}^N$ such that $|R \cdot u| = |R|$. Then, under $R \cdot u$, the counter is synchronized and its value is zero.*

Proof. It is easy to see that each word $u \in L_{\mathsf{reset}}^N$ with $|R \cdot u| = |R|$ cyclically rotates the three tapes of N exactly n times and after each cyclic rotation, either $\iota_{=0}^N$ or ι_{dec}^N is applied. The codomains of both $\iota_{=0}^N$ and ι_{dec}^N do not contain c_0 and thus, we have $R \cdot u \cap \{c_0, c_1, \ldots, c_{n-1}\} = \emptyset$ which is equivalent to saying that the value under $R \cdot u$ is zero. To see that the counter is synchronized, note that applying ι_{sync}^N to a valid configuration preserves the number of elements if and only if the configuration is synchronized. $\qquad\square$

Lemma 15. *Let R be a valid configuration and let $u \in L_{\mathsf{inc}}^N$ such that $|R \cdot u| = |R|$. If v is the value of the counter under R and v' is its value under $R \cdot u$, we have $v' = v + 1 \leqslant 2^n - 1$.*

Proof. Let us first assume that $v < 2^n - 1$. Let $i \in \{0, \ldots, n-1\}$ be minimal such that $c_i \notin R$ and let

$$w = \iota_{\mathsf{sync}}^N (\iota_{\mathsf{dec}}^N \iota_{\mathsf{rotr}}^N \iota_{\mathsf{off}}^N)^i \iota_{\mathsf{inc}}^N (\iota_{\mathsf{off}}^N \iota_{\mathsf{rotr}}^N)^{n-i} \iota_{\mathsf{sync}}^N.$$

We claim that $u = w$. By Lemma 13, it suffices to show that $|R \cdot w| = |R|$. Let us first investigate the instructions operating on S. The word starts with an ι_{sync}^N instruction, each ι_{off}^N instruction is applied after R has been rotated cyclically 1 to $n-1$ times and the second ι_{sync}^N instruction is applied after exactly n cyclic rotations. We deduce $|R \cdot \iota_{\mathsf{sync}}^N| = |R|$ from $|R \cdot u| = |R|$, and thus, the counter is synchronized on both R and on the configuration reached before the last ι_{sync}^N instruction. Moreover, whenever a ι_{off}^N instruction is applied to a configuration R', we have $d_i \in R'$ for some $i \in \{1, \ldots, n-1\}$. Note that the case $v = 2^n - 1$ can be excluded since in order for the ι_{inc}^N instruction to preserve the number of elements in the configuration, it would have to be preceded by at least n $\iota_{\mathsf{rotr}}^N \iota_{\mathsf{off}}^N$-factors and one of those factors would reduce the number of elements.

The instruction ι_{dec}^N is applied exactly once before each of the first i cyclic rotations. Since $\{c_0, c_1, \ldots, c_{i-1}\} \subseteq R$, we have $c_0 \in R \cdot \iota_{\mathsf{sync}}^N (\iota_{\mathsf{dec}}^N \iota_{\mathsf{rotr}}^N \iota_{\mathsf{off}}^N)^j$ for all $j \in \{0, \ldots, i-1\}$. Moreover, since $c_i \notin R$, we have $c_0 \notin R \cdot \iota_{\mathsf{sync}}^N (\iota_{\mathsf{dec}}^N \iota_{\mathsf{rotr}}^N \iota_{\mathsf{off}}^N)^i$ which implies $\bar{c}_0 \in R \cdot \iota_{\mathsf{sync}}^N (\iota_{\mathsf{dec}}^N \iota_{\mathsf{rotr}}^N \iota_{\mathsf{off}}^N)^i$ by Lemma 12. Consequently, the occurrences of ι_{dec}^N and ι_{inc}^N in w do not reduce the number of elements in the configuration. The above observations also show that

$$R \cdot u = R \cdot w = \{c_i\} \cup (R \cap \{c_{i+1}, c_{i+2}, \ldots, c_{n-1}\})$$

which is equivalent to the claim $v' = v + 1$. $\qquad\square$

For the ι_{dec}^N instruction, a symmetric version of the lemma holds.

Lemma 16. *Let R be a valid configuration and let $u \in L_{\text{dec}}^N$ such that $|R \cdot u| = |R|$. If v is the value of the counter under R and v' is its value under $R \cdot u$, we have $v' = v - 1 \geqslant 0$.*

4.4   Main Result

Let $n \in \mathbb{N}$ be an even number. Let T be an n-bit tape with cells $(t_0, t_1, \ldots, t_{n-1})$. The union of T with three $\lceil \log_2 n \rceil$-bit counters P, Q and Z forms a token machine, henceforth referred to as U. A configuration of U is *valid* if it is valid when restricted to each of the three counters.

Informally, the idea of our construction is the following: as in the proof of Theorem 11, we enumerate all $n/2$-element subsets of an n-element set on the tape T. In order to do so with a constant number of generators, this enumeration needs to be done in a very specific way. We say that a word $Y \in \{0, 1\}^*$ is a *successor* of $X \in \{0, 1\}^*$ if there exist $p \in \{0, 1\}^*$, $i \geqslant 1$ and $j \geqslant 0$ such that $X = p01^i0^j$ and $Y = p10^{j+1}1^{i-1}$. For each $m \in \{0, 1, \ldots, n\}$ one can define a sequence of bit strings $(X_0, X_1, \ldots, X_\ell)$ as stated in the following lemma:

Lemma 17. *For all $n \in \mathbb{N}$ and $m \in \{0, 1, \ldots, n\}$, there exists a unique sequence $(X_0, X_1, \ldots, X_\ell)$ such that*

- $X_0 = 0^{n-m}1^m$,
- *for each $k \in \{1, \ldots, \ell\}$, X_k is a successor of X_{k-1} and*
- *X_ℓ does not have a successor.*

The terms of this sequence are pairwise distinct, each term contains exactly m occurrences of the letter 1, and we have $\ell = \binom{n}{m}$ as well as $X_\ell = 1^m0^{n-m}$.

Proof. First observe that if a word $X \in \{0, 1\}^*$ can be factorized as $X = p01^i0^j$ with $p \in \{0, 1\}^*$ and $i \geqslant 1$ and $j \geqslant 0$, then this factorization is unique. As a consequence, the sequence defined above is unique and its terms are pairwise distinct. It is also easy to see that if Y is a successor of X, then X and Y contain the same number of 1's. The remaining two properties $\ell = \binom{n}{m}$ and $X_\ell = 1^m0^{m-n}$ clearly hold if $n = 0$ or $m \in \{0, n\}$.

We now assume $n \geqslant 1$, as well as $m \in \{1, \ldots, n-1\}$, and proceed by induction on n. Let $s \in \{0, \ldots, n\}$ such that $X_0, X_1, \ldots, X_s \in 0\{0, 1\}^{n-1}$ and $X_{s+1}, X_{s+2}, \ldots, X_\ell \in 1\{0, 1\}^{n-1}$. Applying the induction hypothesis to the suffixes of length $n-1$ of $X_0, X_1, \ldots, X_s$, we know that $s = \binom{n-1}{m}$ and $X_s = 01^m0^{(n-1)-m}$. This yields $X_{s+1} = 10^{n-m}1^{m-1}$ and by applying induction again to the suffixes of $X_{s+1}, X_{s+2}, \ldots, X_\ell$, we obtain $\ell - s = \binom{n-1}{m-1}$ as well as $X_\ell = 11^{m-1}0^{(n-1)-(m-1)} = 1^m0^{n-m}$. Note that by Pascal's rule, $\ell = \ell - s + s = \binom{n-1}{m-1} + \binom{n-1}{m} = \binom{n}{m}$ which concludes the proof. $\square$

Note that the sequence corresponds to binary counting and deleting all counter values not having m bits equal 1. Since we are interested in enumerating $n/2$-element subsets, we only consider the case $m = n/2$. Interpreting the bit strings X_k as $n/2$-element subsets of an n-element set, the sequence $(X_0, X_1, \ldots, X_\ell)$ describes our enumeration order. Thus, all configurations appearing in the computation always contain $n/2$ elements when restricted to T. The counter P keeps track of the position of the head on T. It is needed for moving a block of 1-bits as far to the right as possible when transitioning from X_{k-1} to X_k. The volatile counters Q and Z are only used by the following macro that checks whether the bit below the tape head of T is 1.

$$L_{=1} = \iota_{\text{rotr}}^T((\varepsilon \mid \iota_{=0}^T L_{\text{inc}}^Z)\iota_{\text{rotr}}^T L_{\text{inc}}^Q)^* \iota_{\text{val}=n-1}^Q \iota_{\text{val}=n/2}^Z L_{\text{reset}}^Q L_{\text{reset}}^Z.$$

Roughly speaking, a word from $L_{=1}$, which preserves the cardinality of the configuration, rotates the tape T cyclically n times. The counter Q is used to ensure that neither more nor less rotations are performed. After each rotation, except for the last one, the counter Z is increased non-deterministically if the bit under the tape head is 0. Then, the value of Z is checked to be exactly $n/2$. Since we know that the number of 0-bits on T is $n/2$ and since the bit under the tape head cannot contribute to the value of Z, this is only possible if the bit under the tape head is set. More precisely, the following lemma holds.

Lemma 18. *Let R be a valid configuration such that $|R \cap T| = n/2$, the counters P and Q are synchronized and the values of P and Q are zero. Then there exists a word $u \in L_{=1}$ with $|R \cdot u| = |R|$ if and only if $t_0 \in R$. Moreover, if such a word u exists, it is unique and we have $R \cdot u = R$.*

Proof. For $i \in \{0, 1, \ldots, n-1\}$, let $m_i = 1$ if $t_i \notin R$ and let $m_i = 0$ otherwise.

By Lemma 15, the $\iota_{\text{val}=n-1}^Q$ instruction in a word $w \in L_{=1}$ preserves the number of elements in a valid configuration if and only if w contains exactly $n - 1$ occurrences of L_{inc}^Q. Therefore, each word that preserves the number of elements when applied to R contains the instruction ι_{rotr}^N exactly n times. Since each occurrence of L_{inc}^Z is paired with a $\iota_{=0}^T$ instruction, L_{inc}^Z is applied at most m_i times after the i-th rotation, i.e., every program that does not reduce the number of elements when applied to R has the form

$$\iota_{\text{rotr}}^T \prod_{i=1}^{n-1}((\iota_{=0}^T L_{\text{inc}}^Z)^{k_i} \iota_{\text{rotr}}^T L_{\text{inc}}^Q)\iota_{\text{val}=n-1}^Q \iota_{\text{val}=n/2}^Z L_{\text{reset}}^Q L_{\text{reset}}^Z$$

for some $k_i \in \{0, 1\}$ with $k_i \leqslant m_i$. Moreover, the $\iota_{\text{val}=n/2}^Z$ instruction preserves the cardinality of the configuration if and only if the sum of all k_i with $1 \leqslant i \leqslant n-1$ equals $n/2$. Therefore, any choice of values k_i must also satisfy

$$n/2 = \sum_{i=1}^{n-1} k_i \leqslant \sum_{i=1}^{n-1} m_i = n/2 - m_0$$

where the last equality follows from the assumption that $|R \cap T| = n/2$. This is only possible if $m_0 = 0$, i.e., $t_0 \in R$, and $k_i = m_i$ for all $i \in \{1, 2, \ldots, n-1\}$. By letting $k_i = m_i$ in the program above, we obtain the unique word u such that $|R \cdot u| = |R|$. To see that $R \cdot u = R$, note that after n cyclic rotations, the tape T returns to its original state. Moreover, by Lemma 14, both Q and Z are synchronized and have value zero. □

We also let $L_{\mathsf{rotl}} = L_{\mathsf{dec}}^P \iota_{\mathsf{rotl}}^T$ and $L_{\mathsf{rotr}} = L_{\mathsf{inc}}^P \iota_{\mathsf{rotr}}^T$. The language L is now defined as $L = L_{\mathsf{reset}}^P L_{\mathsf{reset}}^Q L_{\mathsf{reset}}^Z (L_{=1} L_{\mathsf{rotr}})^* \iota_{=0}^T L_{\mathsf{rotl}} (\iota_{\mathsf{mvl}}^T (L_1 \mid L_2 \mid L_3))^* \iota_{\mathsf{val}=n-1}^P$ with

$$L_1 = (\iota_{\mathsf{val}=0}^P \mid L_{\mathsf{rotl}} \iota_{=0}^T L_{\mathsf{rotr}}) L_{\mathsf{rotr}} (L_{=1} L_{\mathsf{rotr}})^* \iota_{=0}^T L_{\mathsf{rotl}},$$

$$L_2 = (L_{\mathsf{rotl}} L_{=1})^+ \iota_{\mathsf{val}=0}^P (L_{=1} L_{\mathsf{rotr}})^+ \iota_{=0}^T L_{\mathsf{rotl}},$$

$$L_3 = (L_{\mathsf{rotl}} L_{=1})^+ L_{\mathsf{rotl}} \iota_{=0}^T L_{\mathsf{rotr}} L_{\mathsf{rotr}} (K_1 \mid K_2 K_3^* K_4),$$

$$K_1 = \iota_{=0}^T L_{\mathsf{rotl}} (\iota_{\mathsf{mvr}}^T L_{\mathsf{rotl}})^* \iota_{\mathsf{val}=0}^P,$$

$$K_2 = L_{=1} L_{\mathsf{rotl}} (\iota_{\mathsf{mvr}}^T L_{\mathsf{rotl}})^* \iota_{\mathsf{val}=0}^P L_{\mathsf{rotr}} (\iota_{=0}^T L_{\mathsf{rotr}})^* L_{=1} L_{\mathsf{rotr}},$$

$$K_3 = L_{=1} L_{\mathsf{rotl}} (\iota_{\mathsf{mvr}}^T L_{\mathsf{rotl}})^* L_{\mathsf{rotl}} L_{=1} L_{\mathsf{rotr}} L_{\mathsf{rotr}} (\iota_{=0}^T L_{\mathsf{rotr}})^* L_{=1} L_{\mathsf{rotr}},$$

$$K_4 = \iota_{=0}^T L_{\mathsf{rotl}} (\iota_{\mathsf{mvr}}^T L_{\mathsf{rotl}})^* L_{\mathsf{rotl}} L_{=1} L_{\mathsf{rotr}}.$$

The following lemma is the technical main ingredient for Theorem 20.

Lemma 19. *There exists a valid initial configuration R such that L is deterministic on R. Moreover, there exists a word $u \in L$ of length at least $\binom{n}{n/2}$ such that $|R \cdot u| = |R|$.*

A proof of the lemma can be found in the full version of this paper [5]. Here, we only give a sketch of the arguments. To show that L is deterministic, one can use case distinctions similar to those in the proof of Lemma 13. It then suffices to prove the existence of a word which enumerates the subsets corresponding to the sequence $(X_0, X_1, \ldots, X_\ell)$ defined above. An important invariant is that after each application of a factor from $\iota_{\mathsf{mvr}}^T (L_1 \mid L_2 \mid L_3)$, the tape head points at the leftmost bit of the rightmost 1-block of T. Each such factor replaces the subset corresponding to X_{k-1} by the subset corresponding to X_k on T.

The last missing piece is a component that imposes the language L on the labels of valid computations. To this end, let $\mathcal{A} = (Q, I, \delta, q_0, F)$ be the minimal deterministic automaton of L. We remove the sink state from Q and let all transitions leading to that state be undefined instead. Then, as long as there exists a state which has two ingoing transitions labeled by the same letter, we create a copy of the state and redirect one of the transitions to the copy. When interpreting the letters of I as actions on Q, the tuple (Q, I) then forms a token machine which we call *control unit*. By construction, all instructions are injective.

Theorem 20. *For all $n \in \mathbb{N}$, there exists a token machine with $n + 9 \lceil \log n \rceil + \mathcal{O}(1)$ cells and 32 instructions which admits a maximal progressing computation of length at least $\binom{n}{\lfloor n/2 \rfloor}$.*

Proof. It suffices to prove the theorem for even numbers n. Let V be the union of U and the control unit. Any word, which is not a prefix of a word in L, empties the configuration when applied to the initial configuration $\{q_0\}$ in the control unit. Thus, by taking the union of the initial configuration from Lemma 19 and $\{q_0\}$, we obtain a maximal progressing computation of the desired length in V.

The only instructions required in the construction are ι_{rotl}^T, ι_{rotr}^T, ι_{mvl}^T, ι_{mvr}^T, $\iota_{=0}^T$, $\iota_{\mathsf{val}=0}^P$, $\iota_{\mathsf{val}=n-1}^Q$, $\iota_{\mathsf{val}=n/2}^Z$ and eight additional instructions for each of the three binary counters. Since L is a fixed language, the control unit has c cells for a constant $c \in \mathbb{N}$ (independent of n), and U has $n + 9 \lceil \log n \rceil$ cells: n cells for the tape T and $\lceil \log n \rceil$ cells for each of the three tapes of the three binary counters. Therefore, the number of cells of V is $n + 9 \lceil \log n \rceil + c$. □

Corollary 21. *There exists a sequence of transformation semigroups $(T_n)_{n \in \mathbb{N}}$ with a fixed number of generators such that T_n has n states and the $\mathcal{R}$-height (resp. $\mathcal{L}$-height, $\mathcal{J}$-height) of T_n is in $\Omega(2^n/n^{9.5})$.*

Proof. For the $\mathcal{R}$-height, the result is an immediate consequence of Theorem 20, Propositions 10 and 4. The statement also holds for $\mathcal{J}$-height because every $\mathcal{R}$-chain also is a $\mathcal{J}$-chain; see e.g. [8, Proposition 1.4]. An equivalent statement for the $\mathcal{L}$-height follows from Proposition 5 and the fact that all instructions used in the construction are injective. By Stirling's formula, we have $\binom{n}{n/2} \in \Omega(2^n/n^{0.5})$; see [4,9]. Thus, we obtain the desired bound. Note that the bound in Theorem 20 is for $n + 9 \lceil \log n \rceil + \mathcal{O}(1)$ cells and not just n cells. This yields the factor n^9 in the denominator. □

We can now prove our second main result.

Proof (Theorem 2). In view of Proposition 6 and Proposition 3, the theorem immediately follows from Corollary 21. □

References

1. Brandl, C., Simon, H.U.: Complexity analysis: transformation monoids of finite automata. In: Potapov, I. (ed.) DLT 2015. LNCS, vol. 9168, pp. 143–154. Springer, Cham (2015). doi:10.1007/978-3-319-21500-6_11
2. Carton, O., Michel, M.: Unambiguous Büchi automata. Theoret. Comput. Sci. **297**(1), 37–81 (2003)
3. Eilenberg, S.: Automata, Languages, and Machines, vol. B. Academic Press, New York (1976)
4. Feller, W.: An Introduction to Probability Theory and its Applications, vol. 1. New York, Wiley (1957)
5. Fleischer, L., Kufleitner, M.: Green's Relations in Finite Transformation Semigroups. CoRR, abs/1703.04941 (2017)

6. Ganardi, M., Hucke, D., Lohrey, M.: Querying regular languages over sliding windows. In: FSTTCS 2016, Proceedings, vol. 65. LIPIcs, pp. 18:1–18:14. Dagstuhl Publishing (2016)
7. Holzer, M., König, B.: On deterministic finite automata and syntactic monoid size. Theoret. Comput. Sci. **327**(3), 319–347 (2004)
8. Pin, J.É.: Varieties of Formal Languages. North Oxford Academic, London (1986)
9. Robbins, H.: A remark on Stirling's formula. Am. Math. Monthly **62**, 26–28 (1955)