


The Cayley-Graph of the Queue Monoid: Logic and Decidability

Faried Abu Zaid

Technische Universität Ilmenau, Automata and Logics Group
faried.abu-zaid@tu-ilmenau.de

Chris Köcher

Technische Universität Ilmenau, Automata and Logics Group
chris.koecher@tu-ilmenau.de
 <https://orcid.org/0000-0003-4575-9339>

Abstract

We investigate the decidability of logical aspects of graphs that arise as Cayley-graphs of the so-called queue monoids. These monoids model the behavior of the classical (reliable) fifo-queues. We answer a question raised by Huschenbett, Kuske, and Zetsche and prove the decidability of the first-order theory of these graphs with the help of an - at least for the authors - new combination of the well-known method from Ferrante and Rackoff and an automata-based approach. On the other hand, we prove that the monadic second-order of the queue monoid's Cayley-graph is undecidable.

2012 ACM Subject Classification Theory of computation → Logic, Theory of computation → Models of computation, Information systems → Data structures

Keywords and phrases Queues, Transformation Monoid, Cayley-Graph, Logic, First-Order Theory, MSO Theory, Model Checking

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

Data structures are one of the most important concepts in nearly all areas of computer science. Important data structures are, e.g., finite memories, counters, and (theoretically) infinite Turing-tapes. But the most fundamental ones are stacks and queues. And although these two data structures look very similar as they have got the same set of operations on them (i.e. writing and reading of a letter), they differ from the computability's point of view: if we equip finite automata with both data structures, then the ones with stacks compute exactly the context-free languages (these are the well-known pushdown automata). But if we equip an finite automaton with queues (in literature they are called queue automata, communicating automata, or channel systems) then we obtain a Turing-complete computation model (cf. [2, 3]). This strong model can be weakened with various extensions, e.g., if the queue is allowed to forget some of its contents (cf. [1, 5, 19]) or if letters of low priority can be superseded by letters with higher priority (cf. [9]).

One possible approach to analyze the difference of the behavior of the data structures is to model them as a monoid of transformations. Then, finite memories induce finite monoids, counters induce the integers with addition, stacks induce the polycyclic monoids (cf. [11, 24]), and queues induce the so-called queue monoids which were first introduced in [10]. And while the transformation monoids of the other data structures are very well-understood, we still do not know much about the queue monoid. Further results on the queue monoid (with



© Faried Abu Zaid, Chris Köcher;
licensed under Creative Commons License CC-BY
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and without lossiness) can be found in [14, 15]. Here, we only consider the reliable queue monoids. Concretely, we study the Cayley-graph of this monoid.

Cayley-graphs are a natural translation of finitely generated groups and monoids into graph theory and is a fundamental tool to handle these algebraic constructs in combinatorics, topology, and automata theory. Concretely, these are labeled, directed graphs with labels from a fixed generating set Γ of the monoid \mathcal{M} . Thereby, the elements from \mathcal{M} are the graph's nodes and there is an a -labeled edge (where $a \in \Gamma$) from $x \in \mathcal{M}$ to $y \in \mathcal{M}$ iff $xa = y$ holds in \mathcal{M} . For groups, we already know many results on their Cayley-graphs. For example, the group's Cayley-graph has decidable first-order theory if, and only if, its existential first-order theory is decidable and if, and only if, the group's word problem is decidable [16]. Moreover, a group's Cayley-graph has decidable monadic second-order theory if, and only if, the group is context-free (that is, if the group's word problem is context-free) [16, 20]. Besides these results, Kharlampovich et al. considered in [12] so-called Cayley-graph automatic groups (these are the groups having an automatic Cayley-graph in the sense of [13]) which links to the rich theory of automatic structures.

Unfortunately, there are not that many studies on Cayley-graphs of monoids. In particular, there are monoids with decidable word problem but undecidable existential first-order theory of their Cayley-graph [17, 21]. For finite monoids the Cayley-graphs are finite and, hence, the first- and second-order theories are decidable in polynomial space [7]. For polycyclic monoids the Cayley-graphs are automatic, complete $|A|$ -ary trees (where A is the underlying alphabet) with an additional node every other node is connected with (this is the zero element resp. error state). Therefore, due to [17] the Cayley-graphs monadic second-order theory is decidable (the first-order theory is even in 2EXPSpace by [18]).

In this paper we want to consider logics on the Cayley-graph of the queue monoid. Concretely, we will see that this graph's first-order theory is decidable by giving an primitive recursive (but non-elementary) algorithm which combines two well-known methods from model theory in a (at least for the authors) new way: the method of Ferrante and Rackoff [6] and an automata-based approach. This gives an answer on a question raised by Huschenbett, Kuske, and Zetzsche [10]. There, they conjectured the undecidability of its first-order logic implying that the graph is not automatic in the sense of [13]. Moreover, we will prove the undecidability of the monadic second-order theory with the help of a well-known result from Seese [25].

2 Preliminaries

For $m, n, r \in \mathbb{N}$ we write $m =_r n$ iff $m = n$ or $m, n > r$.

2.1 Graphs

Let $\mathfrak{G} = (V, E)$ be a graph (possibly with some constants $c_1, \dots, c_n \in V$). For $u, v \in V$ we denote by $d^{\mathfrak{G}}(u, v)$ the length of a shortest path from u to v , where we set $d(u, v) = \infty$ whenever u and v are not connected in \mathfrak{G} . For $A \subseteq V$ we denote by $\mathfrak{G}_{\upharpoonright A}$ the induced subgraph of \mathfrak{G} with vertex set A . For a nonempty set $A \subseteq V$ and $r \in \mathbb{N}$ let $\mathcal{N}_r^{\mathfrak{G}}(A)$ denote the r -neighborhood of A (and the constants c_1, \dots, c_n) in \mathfrak{G} , that is $\mathfrak{G}_{\upharpoonright \{u \in V \mid \min\{d(u, v) \mid v \in A \cup \{c_1, \dots, c_n\}\} \leq r\}}$. For a tuple $\vec{u} = (u_1, \dots, u_k) \in V^k$ we will also write $\mathcal{N}_r^{\mathfrak{G}}(\vec{u})$ instead of $\mathcal{N}_r^{\mathfrak{G}}(\{u_1, \dots, u_k\})$.

2.2 Combinatorics on Words

Let Σ be an alphabet. We use \leq to denote the *prefix-relation* and \sqsubseteq for the *suffix-relation* on Σ^* . If $u = vw$ we write $v^{-1}u = w$ and $uw^{-1} = v$. Let $\text{pref}_r(u)$ denote the maximal prefix of u of length at most r . For $u, v \in \Sigma^*$ let $u \sqcap v$ denote the largest suffix of u that is also a prefix of v .

In a first lemma we prove that the complementary prefix and suffix of u resp. v wrt. $u \sqcap v$ can be shortened to words of length at most $2r$ having the same prefixes and suffixes:

► **Lemma 2.1.** *Let $r \in \mathbb{N}$ and $u, v, w \in \Sigma^*$ with $uw \sqcap vw = w$. Then there are words u', v' of length $\mathcal{O}(r)$ such that*

- $\text{suf}_r(uw) = \text{suf}_r(u'w),$
- $\text{suf}_r(wv) = \text{suf}_r(wv'),$
- $\text{pref}_r(wv) = \text{pref}_r(wv'),$
- $u'w \sqcap wv' = w.$

Proof. Set $u' = \text{suf}_r(u)$. Additionally, if $|v| \leq 2r$ set $v' := v$, and otherwise, set $v' := \text{pref}_r(v) \text{suf}_r(v)$. Then the first three equations are obviously satisfied. Now assume $u'w \sqcap wv' \neq w$, i.e., there is $w' \in \Sigma^*$ with $|w'| > |w|$, $w' \leq wv'$, and $w' \sqsubseteq u'w$. Since $|u'w| \leq r + |w|$ we have $w' \leq w \text{pref}_r(v) \leq wv$. Additionally, we have $w' \sqsubseteq u'w \sqsubseteq uw$ implying $|uw \sqcap vw| \geq |w'| > |w|$. This is a contradiction to the definition of w . ◀

A *period* of a word u is a word v such that $u \leq v^\omega$. Obviously every word u has a unique smallest period, which we denote by \sqrt{u} . The *left-exponent* of u in v is the largest number n such that $v = u^n w$, and it is denoted by $\text{lexp}(u, v)$. The *right-remainder*, $v \bmod u$, of v with respect to u is defined as $(u^{\text{lexp}(u, v)})^{-1}v$, that is the unique w such that $v = u^{\text{lexp}(u, v)}w$. In particular we have $v = \sqrt{v}^{\text{lexp}(\sqrt{v}, v)}(v \bmod \sqrt{v})$ for every $v \in \Sigma^*$. A word u is *primitive* if there is no v with $|v| < |u|$ and $u = v^n$ for some $n \in \mathbb{N}$. For $u, v \in \Sigma^*$ let $u\Delta v = (y, z)$, where y, z are minimal such that there exists an x with $u = xy$ and $v = xz$. For $\vec{v}, \vec{w} \in (\Sigma^*)^k$ let $\vec{v}\Delta\vec{w} = (v_1\Delta w_1, \dots, v_k\Delta w_k) \in (\Sigma^*)^{2k}$ and $|\vec{w}| := \sum_{i=1}^k |w_i|$.

► **Definition 2.2.** Let $u \in \Sigma^*$ be a word. A *canon-decomposition* of u is a sequence of words $\varepsilon = u_0, u_1, \dots, u_n = u$ such that for all $0 \leq i < n$ it holds that $u_i \leq u_{i+1}$ and $u_i \sqsubset u_{i+1}$ ($u_i \leq u_{i+1}$ for short). A canon-decomposition u_0, u_1, \dots, u_n is *complete* if there is no $1 \leq i < n$ and $v \in \Sigma^*$ with $u_i \leq v \leq u_{i+1}$.

► **Lemma 2.3.** *Every word $w \in \Sigma^*$ has a unique complete canon-decomposition.*

Proof. Obviously, every word w possesses at least one complete canon-decomposition. Now suppose $\vec{u} = (u_0, \dots, u_m)$ and $\vec{v} = (v_0, \dots, v_n)$ are two distinct complete canon-decompositions of a word $w \in \Sigma^*$. W.l.o.g. assume that $n \leq m$. We claim that there is an $0 \leq i \leq n$ with $u_i \neq v_i$. Because otherwise it follows from $w = u_n = v_n$ that \vec{u} and \vec{v} have the same length n and this, in turn, implies that they are identical since $u_i = v_i$ for all $0 \leq i \leq n$. Now choose the smallest i between 0 and n such that $u_i \neq v_i$. As $u_0 = \varepsilon = v_0$ it holds that $i > 0$. Hence u_{i-1}, v_{i-1} are defined and $u_{i-1} = v_{i-1}$. Since $u_i, v_i \leq w$ and $u_i \neq v_i$ it must be the case that $|u_i| \neq |v_i|$. Again w.l.o.g. assume that $|u_i| < |v_i|$. Then $u_i, v_i \leq w$, $u_i, v_i \sqsubseteq w$ and $|u_i| < |v_i|$, which implies $u_i \sqsubset v_i$. Therefore $v_{i-1} = u_{i-1} \sqsubset u_i \sqsubset v_i$ in contradiction to the completeness of \vec{v} ! ◀

► **Example 2.4.** The complete canon-decomposition of *ababa* is $(\varepsilon, a, aba, ababa)$.

From the complete canon-decomposition of a word w we derive the so called skeleton of w containing the inner words v of all canons uvu in w .

► **Definition 2.5.** Let $w \in \Sigma^*$ and $\vec{w} = (w_0, \dots, w_n)$ be the complete canon-decomposition of w . The r -skeleton of w , denoted by $\mathcal{S}_r(w)$, is the word of length n over the alphabet $\Gamma = \Sigma^{\leq r}$ with $\mathcal{S}_r(w)[i] = \text{pref}_r(w_i^{-1}w)$ for each $0 \leq i \leq n-1$. Note that $w_i^{-1}w$ is always defined since $w_i \leq w$.

► **Example 2.6.** Let $u = bababa$ and $v = ababab$. Then $u \sqcap v = ababa$ and the complete canon-decomposition of $u \sqcap v$ is $(\varepsilon, a, aba, ababa)$. The 2-skeleton of $u \sqcap v$ is the word depicted below.

$$ab \longrightarrow ba \longrightarrow ba$$

By definition the i -th element of the canon decomposition of $u \sqcap v$ corresponds to the i -th letter of the skeleton. We will use this correspondence to translate back and forth between an Ehrenfeucht-Fraïssé game played on the Cayley-graph of a queue monoid and games played on certain skeletons which are derived from the game played on the Cayley-graph.

► **Lemma 2.7.** Let $r \in \mathbb{N}$, $w \in \Sigma^*$ and $n \in \mathbb{N}$ be the length of $\mathcal{S}_r(w)$. Then a word $v \in \Sigma^*$ can be constructed from w such that $|v| = \mathcal{O}(2^{nr})$ and $\mathcal{S}_r(w) = \mathcal{S}_r(v)$.

Proof. Let $\vec{w} = (w_0, \dots, w_n)$ be the complete canon-decomposition of w . At first, assume $|\mathcal{S}_r(w)[n-1]| < r$ (i.e., the last component is small). Then there are two possibilities: on the one hand $w = w_{n-1}xw_{n-1}$ and $|xw_{n-1}| < r$. In this case we have $|w| < 2r = \mathcal{O}(2^{nr})$. On the other hand we have $w = xw_{n-1} = w_{n-1}y$ where $|x| = |y| < \min\{|w_{n-1}|, r\}$, i.e., the prefix and the suffix w_{n-1} overlap in w_n . Then it is easy to see that x is a period of w_{n-1} and of w_n . Concretely, there is a prefix p of x and a number $k \in \mathbb{N}$ such that $w = x^k p$ and $w_{n-1} = x^{k-1} p$. In particular, all word $x^i p$ with $1 \leq i \leq k$ are borders of w which implies $k \leq n$. Hence we have $|w| \leq |x| \cdot (k+1) \leq r \cdot (n+1) = \mathcal{O}(2^{nr})$. Therefore, in both cases we are ready and we can assume $|\mathcal{S}_r(w)[n-1]|$ from now on.

We construct v inductively as follows: We set $v_0 := \varepsilon$. Now let $a, b \in \Sigma$ be distinct with $\mathcal{S}_r(w)[0] \in a\Sigma^*$. Then $x \sqsubseteq \mathcal{S}_r(w)[0]b^{2n+r}$ implies $x = \varepsilon$. Hence, we set, for $0 \leq i < n$, $v_{i+1} := v_i x_i v_i$ where $x_i = \mathcal{S}_r(w)[i] b^{n-i} a^i b^{n+r}$. Finally, we set $v := v_n$.

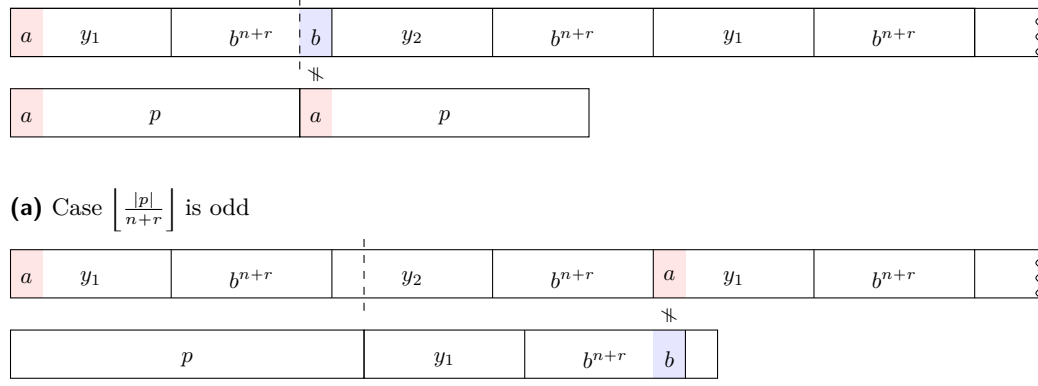
Before we can prove $\mathcal{S}_r(w) = \mathcal{S}_r(v)$ we need to prove the following two properties of (v_0, \dots, v_n) :

- (a) For each $0 \leq i \leq n$ $\sqrt{v_{i+1}} = v_i x_i$ and
- (b) $\vec{v} = (v_0, \dots, v_n)$ is a complete canon-decomposition of v .

Proof of (a). We observe that $v_i x_i$ is a period of v_{i+1} and we prove by induction on $0 \leq i \leq n$ that this period is minimal. For $i = 0$ this is trivial since $v_1 \in a\Sigma^{r-1}b^{2n+r}$ and $a \neq b$. So now let $i > 0$. We suppose that there is a period p of v_{i+1} with $|p| < |v_i x_i|$. Then, for $y_j := x_j(b^{n+r})^{-1}$ for $0 \leq j \leq i$, the word v_{i+1} is an alternation of words y_j and b^{n+r} which are all of length $r+n$. Note that by construction we have $y_j \neq b^{n+r}$ (since each y_j contains at least one a) as well as $y_j \neq y_k$ if $j \neq k$ for each $0 \leq j, k \leq i$. Additionally, each second occurrence of a y_j -block is y_1 . We now consider two cases:

First, assume that $|p|$ is not a divisor of $n+r$. If $|p| < n+r$ then the distance between each two occurrences of a in p^ω is at most $|p| < n+r$ but v_{i+1} contains at least one b^{n+r} -block. Hence, we have $|p| > n+r$. If $\lfloor \frac{|p|}{n+r} \rfloor$ is odd (cf. Fig. 1a), p starts with a and ends in a block of the form b^{n+r} , but does not contain all of these $n+r$ many b 's. Since p start with an a , a first repetition of p this first a is different from the b at this position in v_{i+1} , i.e., p is not a period of v_{i+1} . Otherwise, if $\lfloor \frac{|p|}{n+r} \rfloor$ is even (cf. Fig. 1b), then the prefix of $p^{-1}v_{i+1}$ of length $|p|$ contains at most one y_1 -block and this overlaps with a b^{n+r} -block. Hence, there

171 is a position in the first repetition of p containing a b which is different from the a at this
 172 position in v_{i+1} .



■ **Figure 1**

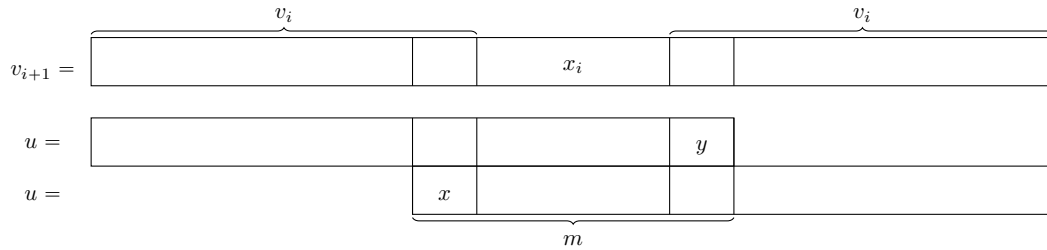
173 Now, assume $|p|$ is a divisor of $n + r$. Then we can understand the blocks of length
 174 $n + r$ as letters of the alphabet $\{b^{n+r}, y_1, \dots, y_i\}$. Since there is no y_i -block in v_i we have
 175 $|p| \geq |v_i y_i|$. Since p starts with y_1 and y_i is followed by b^{n+r} , p has length at least $|v_i x_i|$.

176 *Proof of (b).* By construction, it is easy to see that $\vec{v} = (v_0, \dots, v_n)$ is a canon-decomposition
 177 of $v = v_n$. We prove now by induction on $0 \leq i < n$ that (v_0, \dots, v_{i+1}) is a complete canon-
 178 decomposition of v_i . The case $i = 0$ is easy to verify since $v_1 \in aA^{r-1}b^{2n+r}$. So, let $i \geq 1$.
 179 Assume there is $u \in \Sigma^*$ with $v_i \leq u \leq v_{i+1}$. Let u be of minimal length satisfying this
 180 inequality. Then there are two possible cases:

181 First, suppose $|u| \geq |v_i x_i|$ holds, i.e., the prefix and suffix u overlap in v_i and the overlap
 182 contains at most x_i (cf. Fig. 2). Let $x, y \in \Sigma^*$ such that $u = x x_i v_i = y$. Then we have
 183 $|x| = |y|$ and $m := x x_i y \leq u$. Hence, by minimality of u we have $|m| \leq |v_i|$ and therefore, by
 184 induction hypothesis, $m = v_k$ for some $0 < k \leq i$. This implies

$$185 \quad v_{k-1} x_{k-1} v_{k-1} = v_k = m = x x_i y.$$

186 Since $|x| = |y|$ and $|x_i| = |x_{k-1}|$ we have $x_i = x_{k-1}$, which is a contradiction to the
 187 construction of the x_i 's.



■ **Figure 2**

188 Now, suppose $|u| < |v_i x_i|$. If $|u| \geq \frac{|v_{i+1}|}{2}$ (i.e., the prefix and suffix u in v_i overlap) then

there is a word $m \in \Sigma^*$ such that $m \leq u$ holds. Hence, by minimality of u and by induction hypothesis we have $m = v_k$ for some $0 \leq k \leq i$. Since $|m| < |x_i| = |x_1|$ we have $m = \varepsilon$, i.e., we have $|u| = \frac{|v_{i+1}|}{2}$.

Suppose $|u| \leq \frac{|v_{i+1}|}{2}$ (i.e., the prefix and suffix u in v_i do not overlap). Then there is a word $p \in \Sigma^*$ such that $v_{i+1} = pu$. Since u is a prefix of v_{i+1} and $|p| > \frac{|v_{i+1}|}{2}$, u also is a prefix of p . Hence, p is a period of v_{i+1} and we have

$$|p| = |v_{i+1}| - |u| < |v_{i+1}| - |v_i| = |v_i x_i|.$$

This is a contradiction to property a stating that $v_i x_i$ is the minimal period of v_{i+1} .

So, in both cases we have seen that there is no $v_i \leq u \leq v_{i+1}$, i.e., (v_0, \dots, v_{i+1}) is a complete canon-decomposition.

Finally, let $0 \leq i < n$. Then we have

$$\mathcal{S}_r(v)[i] = \text{pref}_r(v_i^{-1}v) = \text{pref}_r(\mathcal{S}_r(w)[i]s) = \mathcal{S}_r(w)[i]$$

for some $s \in \Sigma^*$, i.e., $\mathcal{S}_r(v) = \mathcal{S}_r(w)$. ◀

Let $V \in (\Sigma^{\leq r})^*$ be the r -skeleton of some word $w \in \Sigma^*$. We call the word $v \in \Sigma^*$ constructed in the proof of Lemma 2.7 the *canonical r -instantiation of V* .

3 Queue Monoid and its Cayley-Graph

3.1 Definition of the Monoid

The queue monoid models the behavior of a (reliable) fifo-queue whose entries come from an alphabet A . Consequently, the state of a queue is a word from A^* . The basic actions of our queue are writing of the symbol $a \in A$ of the queue (denoted by a) and reading the symbol $a \in A$ from the queue (denoted by \bar{a}). Thereby, \bar{A} is a disjoint copy of A containing all reading actions \bar{a} and $\Sigma := A \cup \bar{A}$ is the set of all basic actions. To simplify notation, for a word $u = a_1 a_2 \dots a_n \in A^*$ we write \bar{u} for the word $\bar{a}_1 \bar{a}_2 \dots \bar{a}_n$.

Formally, the action $a \in A$ appends the letter a to the state of the queue and the action $\bar{a} \in \bar{A}$ tries to cancel the letter a from the beginning of the current state of the queue. Thereby, if the state does not start with this symbol, the queue will end up in an error state which we denote by \perp . Note that in contrast to (partially) lossy queues which we considered in [14, 15], these queues cannot forget any part of their content. Hence, these ideas lead to the following definition:

► **Definition 3.1.** Let $\perp \notin A^*$. The function $\circ: (A^* \cup \{\perp\}) \times \Sigma^* \rightarrow (A^* \cup \{\perp\})$ is defined for each $q \in A^*$, $a, b \in A$, and $u \in \Sigma^*$ as follows:

- (1) $q \circ \varepsilon = q$
- (2) $q \circ au = qa \circ u$
- (3) $bq \circ \bar{a}u = \begin{cases} q \circ u & \text{if } a = b \\ \perp & \text{otherwise} \end{cases}$
- (4) $\varepsilon \circ \bar{a}u = \perp \circ u = \perp$

With the help of this function we may now identify sequences of actions that are acting equally. This is finally used to define the monoid of queue actions.

► **Definition 3.2.** Let $u, v \in \Sigma^*$. Then u and v act *equally* (denoted by $u \equiv v$) if $q \circ u = q \circ v$ holds for each $q \in A^*$.

Since $q \circ uv = (q \circ u) \circ v$, the resulting relation \equiv is a congruence on the free monoid Σ . Hence, the quotient $\mathcal{Q}(A) := \Sigma^* / \equiv$ is a monoid which we call the *monoid of queue actions* or for short *queue monoid*.

Note that the queue monoids $\mathcal{Q}(A)$ for alphabets A of different size are not isomorphic. Though, all of the following results hold for any alphabet A with $|A| \geq 2$. Hence, we may fix an arbitrary alphabet A from now on and write \mathcal{Q} instead of $\mathcal{Q}(A)$.

► **Remark.** Let $A = \{a\}$ be a singleton. Then a queue on this alphabet acts like a partially blind counter since $a^n \circ a = a^{n+1}$ and $a^{n+1} \circ \bar{a} = a^n$. In other words, $\mathcal{Q}(\{a\})$ is the bicyclic semigroup.

3.2 Basic Properties

Now, we want to recall some basic properties considering the equivalence relation \equiv . The first important fact expresses the equivalence in terms of some commutations of write and read actions under certain contexts.

► **Theorem 3.3** ([10, Theorem 4.3]). *The equivalence relation \equiv is the least congruence on the free monoid Σ^* satisfying the following equations for all $a, b \in A$:*

$$(1) \quad a\bar{b} \equiv \bar{b}a \text{ if } a \neq b$$

$$(2) \quad a\bar{a}\bar{b} \equiv \bar{a}a\bar{b}$$

$$(3) \quad ba\bar{a} \equiv b\bar{a}a$$

A very frequently used notation is the following: the *projections to write and read actions*, resp., are defined as $\text{wrt}, \text{rd}: \Sigma^* \rightarrow A^*$ by $\text{wrt}(a) = \text{rd}(\bar{a}) = a$ and $\text{wrt}(\bar{a}) = \text{rd}(a) = \varepsilon$ for all $a \in A$. In other words, $\text{wrt}(u)$ can be derived from u by deletion of all read actions and $\text{rd}(u)$ can be obtained from u by deletion of all the write actions and by suppression of the overlines. Due to Theorem 3.3 all words contained in a single equivalence class of \equiv have the same projections. Hence we use them for equivalence classes as well. Though, equality of these projections of two words does not imply equivalence of these words. For example, $u = \bar{a}a$ and $v = a\bar{a}$ have the same projections $\text{wrt}(u) = \text{rd}(u) = a = \text{wrt}(v) = \text{rd}(v)$ but are not equivalent according to Theorem 3.3.

The non-equivalence of the two words above is very easy to prove. Also (non-)equivalence of two arbitrary words is decidable in polynomial time: for this purpose we compute normal forms of the equivalence classes of \equiv . We do this by ordering the equations from Theorem 3.3 from left to right resulting in a terminating and confluent semi-Thue system \mathcal{R} [10, Lemma 4.1]. Then, for any word $u \in \Sigma^*$ there is a unique, irreducible word $\text{nf}(u)$ with $u \rightarrow^* \text{nf}(u)$, the so-called *normal form* of u resp. of its equivalence class $[u]$. In this word $\text{nf}(u)$ the read actions from u are moved to the left as far as the equations from above allow.

► **Example 3.4.** Let $a, b \in A$ with $a \neq b$ and $u = abb\bar{a}\bar{b}$. Then we have

$$abb\bar{a}\bar{b} \xrightarrow{(1)} ab\bar{a}b\bar{b} \xrightarrow{(1)} a\bar{a}b\bar{b}\bar{b} \xrightarrow{(3)} a\bar{a}b\bar{b}b.$$

Since we cannot apply any rule from Theorem 3.3 anymore, we have $\text{nf}(u) = a\bar{a}b\bar{b}b$.

From the definition of \mathcal{R} we obtain that a word is in normal form if it starts with a sequence of read operations followed by an alternating sequence of write and read actions, where all of the read actions \bar{a} appear straight behind the write action a . Finally, the normal form ends with a sequence of write actions. Concretely, the set of all normal forms is

$$\text{NF} := \{\text{nf}(u) \mid u \in \Sigma^*\} = \bar{A}^* \{a\bar{a} \mid a \in A\}^* A^*.$$

Let $u \in \Sigma^*$. Then the normal form $\text{nf}(u)$ is uniquely defined by three words $u_1, u_2, u_3 \in A^*$ such that $\text{nf}(u) = \overline{u_1}a_1\overline{a_1} \dots a_n\overline{a_n}u_3$ where $u_2 = a_1 \dots a_n$. Thereby, we denote the word u_1 by $\lambda(u)$, the word u_2 by $\mu(u)$, and u_3 by $\varrho(u)$. Hence, we can define the *characteristics* of u ($[u]$, resp.) by the triple

$$\chi(u) := (\lambda(u), \mu(u), \varrho(u)).$$

Hence, from these characteristics $\chi(u)$ we can obtain the projections of u on its write and read actions as well: $\text{wrt}(u) = \mu(u)\varrho(u)$ and $\text{rd}(u) = \lambda(u)\mu(u)$.

From now on, we will use these characteristics to represent the elements of \mathcal{Q} . In other words, we may understand \mathcal{Q} as a triple of words (i.e., $(A^*)^3$) with a special type of concatenation which is described in the following Theorem:

► **Theorem 3.5** ([10, Theorem 5.3]). *Let $u, v \in \Sigma^*$. Then*

$$\chi(uv) = (\lambda(u)r, s, t\varrho(v))$$

where $s = \mu(u)\text{rd}(v) \sqcap \text{wrt}(u)\mu(v)$, $rs = \mu(u)\text{rd}(v)$, and $st = \text{wrt}(u)\mu(v)$. ◀

In other words, the multiplication of two words $u, v \in \Sigma^*$ can be understood as follows: at first we move the read actions from $\text{rd}(v)$ to the left such that each of its letters is directly preceded by exactly one write actions. If this is not possible (because $\lambda(v)$ is longer than $\varrho(u)$) we move the letters from $\mu(u)\lambda(v)$ to the left until there is an alternating word of write and read actions. Now, if there is an infix $a\bar{b}$ with $a \neq b$ all of these read actions move one position to the left. We iterate this last step until there is no such infix. It is easy to see, that the new alternating word contains equal subsequences of write and read actions, respectively. Thereby, the read actions are the longest suffix of $\overline{\mu(u)\text{rd}(v)}$ and the write actions the longest prefix of $\text{wrt}(u)\mu(v)$ such that the equality of these subsequences holds.

3.3 The Monoid's Cayley-Graph

In this subsection we first recall the definition of Cayley-graphs for arbitrary, finitely generated monoids. Afterwards, we give some common properties as well as some special characteristics of the queue monoid's Cayley-graph.

► **Definition 3.6.** Let \mathcal{M} be a monoid generated by a finite set $\Gamma \subset \mathcal{M}$. The (*right*) *Cayley-graph* of \mathcal{M} is the edge-labeled, directed graph $\mathfrak{C}(\mathcal{M}, \Gamma) := (\mathcal{M}, (E_a)_{a \in \Gamma})$ with

$$E_a = \{(x, y) \in \mathcal{M} \mid y = xa\}$$

for any $a \in \Gamma$.

Similar to the right Cayley-graph, we may define the *left Cayley-graph* of \mathcal{M} as the edge-labeled, directed graph $\mathfrak{L}\mathfrak{C}(\mathcal{M}, \Gamma) = (\mathcal{M}, (F_a)_{a \in \Gamma})$ with $F_a = \{(x, y) \in \mathcal{M} \mid y = ax\}$ for any $a \in \Gamma$.

► **Remark.** There is a strong relation between left and right Cayley-graphs of a monoid and Green's relations which are first introduced and studied in [8]. Recall that $x\mathcal{R}y$ iff $x\mathcal{M} = y\mathcal{M}$ for any $x, y \in \mathcal{M}$ and, similarly, $x\mathcal{L}y$ iff $\mathcal{M}x = \mathcal{M}y$. Then by [22, Proposition V.1.1] we have $x\mathcal{R}y$ ($x\mathcal{L}y$) if, and only if, x is strongly connected to y in $\mathfrak{C}(\mathcal{M}, \Gamma)$ ($\mathfrak{L}\mathfrak{C}(\mathcal{M}, \Gamma)$, resp.).

309 The concrete shape of the Cayley-graph of a monoid heavily depends on the chosen
 310 set of generators. For example, $\{-1, 1\}$ and $\{-2, 3\}$ are generating sets of $(\mathbb{Z}, +)$, but the
 311 resulting Cayley-graphs are not isomorphic (even if we remove the labels). Though, the
 312 chosen generating set has no influence on decidability and complexity of the FO and MSO
 313 theory of the Cayley-graph since the both problems are logspace reducible on each other
 314 (which we denote by \approx_{\log}):

315 ▶ **Proposition 3.7** ([17, Proposition 3.1]). *Let Γ_1 and Γ_2 be two finite generating sets of the*
 316 *monoid \mathcal{M} . Then*

- 317 (1) $\text{FOTh}(\mathfrak{C}(\mathcal{M}, \Gamma_1)) \approx_{\log} \text{FOTh}(\mathfrak{C}(\mathcal{M}, \Gamma_2))$ and
 318 (2) $\text{MSOTh}(\mathfrak{C}(\mathcal{M}, \Gamma_1)) \approx_{\log} \text{MSOTh}(\mathfrak{C}(\mathcal{M}, \Gamma_2))$. ◀

319 From now on we only consider the Cayley-graph of the queue monoid \mathcal{Q} . To simplify
 320 notation we write \mathfrak{C} instead of $\mathfrak{C}(\mathcal{Q}, \Sigma)$ and $\mathfrak{L}\mathfrak{C}$ instead of $\mathfrak{L}\mathfrak{C}(\mathcal{Q}, \Sigma)$. First we prove some
 321 properties of \mathfrak{C} and $\mathfrak{L}\mathfrak{C}$.

322 ▶ **Proposition 3.8.** *The following statements hold:*

- 323 (1) $\text{FOTh}(\mathfrak{C}) \approx_{\log} \text{FOTh}(\mathfrak{L}\mathfrak{C})$ and $\text{MSOTh}(\mathfrak{C}) \approx_{\log} \text{MSOTh}(\mathfrak{L}\mathfrak{C})$.
 324 (2) \mathfrak{C} is an acyclic graph with root $[\varepsilon]$.
 325 (3) \mathfrak{C} has unbounded (in-)degree.

326 **Proof.** At first, we prove (1). Let the duality function $\delta: \Sigma^* \rightarrow \Sigma^*$ be defined as follows:

327
$$\delta(\varepsilon) = \varepsilon, \delta(au) = \delta(u)\bar{a}, \text{ and } \delta(\bar{a}u) = \delta(u)a$$

328 for all $u \in \Sigma^*$ and $a \in A$. In other words, δ reverses the order of the actions and inverts
 329 writing and reading of a letter a . From [10, Proposition 3.4] we know $u \equiv v$ iff $\delta(u) \equiv \delta(v)$.
 330 Hence, δ is an automorphism on \mathcal{Q} and $(p, q) \in E_\alpha$ iff $(\delta(p), \delta(q)) \in F_{\delta(\alpha)}$ for all $p, q \in \mathcal{Q}$ and
 331 $\alpha \in \Sigma$. Let $\varphi \in \text{FO}[(E_\alpha)_{\alpha \in \Sigma}]$ ($\varphi \in \text{MSO}[(E_\alpha)_{\alpha \in \Sigma}]$, resp.). We construct φ' by replacing any
 332 atom “ $E_\alpha(x, y)$ ” in φ by “ $F_{\delta(\alpha)}(x, y)$ ”. Then

333
$$\mathfrak{C} \models \varphi[q_1, \dots, q_k] \iff \mathfrak{L}\mathfrak{C} \models \varphi'[\delta(q_1), \dots, \delta(q_k)]$$

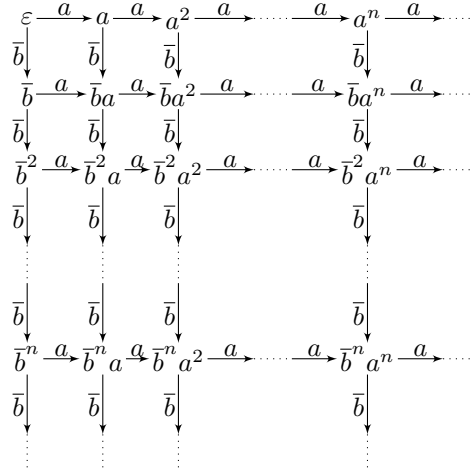
334 for any $q_1, \dots, q_k \in \mathcal{Q}$. In particular, $\varphi \in \text{FOTh}(\mathfrak{C})$ iff $\varphi' \in \text{FOTh}(\mathfrak{L}\mathfrak{C})$ (resp. $\varphi \in \text{MSOTh}(\mathfrak{C})$
 335 iff $\varphi' \in \text{MSOTh}(\mathfrak{L}\mathfrak{C})$). Finally, the converse reduction is symmetric to the one described
 336 above.

337 Now, we prove (2). Due to [10, Corollary 4.7] we have $p\mathcal{R}q$ iff $p = q$ for all $p, q \in \mathcal{Q}$.
 338 Then, by the remark above $p, q \in \mathcal{Q}$ are strongly connected iff $p = q$, i.e., there are no cycles
 339 in \mathfrak{C} .

340 Next, to prove (3) let $n \in \mathbb{N}$ and $a, b \in A$ with $a \neq b$. Set $w_k = \bar{a}^k(a\bar{a})^{n-k}a^k$ for any
 341 $0 \leq k \leq n$. Then $w_k \equiv w_\ell$ (i.e. $[w_k] = [w_\ell]$) iff $k = \ell$ for any $0 \leq k, \ell \leq n$. By Theorem 3.5
 342 we have $\chi(w_k\bar{b}) = (a^n b, \varepsilon, a^n)$, i.e. $w_k\bar{b} \equiv w_\ell\bar{b}$ for any $0 \leq k, \ell \leq n$. Hence, we have
 343 $([w_k], [\bar{a}^n b a^n]) \in E_{\bar{b}}$ for all $0 \leq k \leq n$, i.e., the node $[\bar{a}^n b a^n]$ has in-degree $> n$. ◀

344 By \mathfrak{G}_n we denote the $n \times n$ -grid for $n \in \mathbb{N}$. This is an undirected graph with n^2 many
 345 nodes which we denote by $v_{i,j}$ for any $1 \leq i, j \leq n$. Thereby, we have an edge between $v_{i,j}$
 346 and $v_{k,\ell}$ if, and only if, $|j - \ell| + |i - k| = 1$ holds. Additionally, for a Γ -labeled, directed
 347 graph $\mathfrak{G} = (V, (E_a)_{a \in \Gamma})$ we denote the unlabeled and undirected version by $\text{ud}(\mathfrak{G}) = (V, E)$.
 348 Here, we have an edge $(v, w) \in E$ if, and only if, there is an $a \in \Gamma$ such that $(v, w) \in E_a$ or
 349 $(w, v) \in E_a$. Then, in $\text{ud}(\mathfrak{C})$ we can find \mathfrak{G}_n for any $n \in \mathbb{N}$:

350 ▶ **Proposition 3.9.** \mathfrak{G}_n is an induced subgraph of $\text{ud}(\mathfrak{C})$ for any $n \in \mathbb{N}$.



■ **Figure 3** \mathfrak{C} restricted to the nodes reachable by a - and \bar{b} -edges, only.

Proof. Let $a, b \in A$ be distinct. Then the submonoid \mathcal{M} of \mathcal{Q} generated by a and \bar{b} is the free commutative monoid on $\{a, \bar{b}\}$ by Theorem 3.3(1). Its Cayley-graph $\mathfrak{C}(\mathcal{M}, \{a, \bar{b}\})$ is an infinite grid with labeled, directed edges (see Fig. 3). Then, \mathfrak{G}_n is an induced subgraph of $\text{ud}(\mathfrak{C}(\mathcal{M}, \{a, \bar{b}\}))$. Since in \mathfrak{C} there are no edges with labels other than a or \bar{b} between the nodes from \mathcal{M} , $\text{ud}(\mathfrak{C}(\mathcal{M}, \{a, \bar{b}\}))$ is an induced subgraph of $\text{ud}(\mathfrak{C})$ as well implying our claim. ◀

With the help of a famous result from Seese (cf. [25]), we may now prove the undecidability of the monadic second-order theory of the queue monoid's Cayley-graph.

► **Corollary 3.10.** *MSOTh(\mathfrak{C}) is undecidable.*

Proof. Due to [23] each planar graph is a minor of some grid \mathfrak{G}_n . Since each \mathfrak{G}_n is an induced subgraph of $\text{ud}(\mathfrak{C})$ by Proposition 3.9, each planar graph is minor of an induced subgraph of $\text{ud}(\mathfrak{C})$. Hence, by [25, Theorem 5] $\text{MSOTh}(\text{ud}(\mathfrak{C}))$ is undecidable. Since $\text{ud}(G)$ is interpretable in $\text{FOTh}(\mathfrak{C})$, $\text{MSOTh}(\mathfrak{C})$ is undecidable as well. ◀

4 Decidability of the FO-Theory

In the following we denote the Cayley-graph of the queue monoid \mathcal{Q} induced by A by $\mathfrak{C} = (C, (E_\alpha)_{\alpha \in \Sigma})$. For $a, b \in C$ let $a\Delta b = (\text{rd}(a), \text{wrt}(a))\Delta(\text{rd}(b), \text{wrt}(b))$ and we call $|a\Delta b|$ the (Δ) -distance of a and b .

► **Definition 4.1.** Let V be an r -skeleton. We say that $a \in C$ is *compatible* with V if V has an instantiation v such that $\text{rd}(a) = \text{rd}(v)x$ for some $x \in A^{\leq r}$ and $|\text{wrt}(a)\Delta\text{wrt}(v)| \leq r$.

Intuitively, a being compatible to an r -skeleton V means that we can obtain an element a' with r -skeleton V by deleting up to r many read actions and modifying the write actions arbitrarily up to distance r . We use this notion in order to translate elements of the Cayley-graph into positions of an r -skeleton.

► **Definition 4.2.** For $a \in C$ with $|\mu(a)| \geq r$ let $\text{cpr}_r(a)$ be the element a' with $\text{wrt}(a') = \text{wrt}(a)$, $\text{rd}(a') = \text{rd}(a) \text{suf}_r(\text{rd}(a))^{-1}$, and $\mu(a') = \mu(a) \text{suf}_r(\mu(a))^{-1}$.

We describe the way, in which we associate positions in an r -skeleton with elements of C and vice versa.

Definition 4.3. Let $a, b \in C$ and let U and V be the $3r$ -skeletons of $\text{cpr}_{2r}(a)$ and $\text{cpr}_{2r}(b)$ respectively. If we suppose that (p_1, \dots, p_k) are positions in V and (q_1, \dots, q_k) are positions in U such that $(U, p_1, \dots, p_k) \equiv_\ell (V, q_1, \dots, q_k)$ for some $\ell \geq 1$. For $a' \in C$ with $|a' \Delta a| \leq r$ we associate a position p_{k+1} in U as follows: Let (u_1, \dots, u_m) be the complete canonical decomposition of $\text{rd}(\text{cpr}_{2r}(a))$ and (v_1, \dots, v_n) . As a' has distance at most r from a we have that $\text{rd}(a') = \text{rd}(\text{cpr}_{2r}(a))x$ for some $x \in A^{\leq 2r}$. Therefore there is an $i \leq m$ such that $\mu(a') = u_i x$. Then i is the position that is associated with a' .

Now let q_{k+1} be such that $(U, p_1, \dots, p_{k+1}) \equiv_{\ell-1} (V, q_1, \dots, q_{k+1})$ we associate an element b' with q_{k+1} as follows: Let b' be the element with $\text{rd}(b') = \text{rd}(\text{cpr}_{2r}(b))u_{p_{k+1}}^{-1}\mu(a')$, $\text{wrt}(b')\Delta\text{wrt}(\text{cpr}_r(b)) = \text{wrt}(a')\Delta\text{wrt}(\text{cpr}_{2r}(a))$, and $\mu(b') = v_{q_{k+1}}u_{p_{k+1}}^{-1}\mu(a')$. Note that b' is well defined since $V[j]$ is labeled by $\text{pref}_{2r+2}(u_i^{-1}\mu(a))$. Therefore $v_j \text{pref}_{2r+1}(v_i^{-1}\mu(a))$ is a prefix of $\text{wrt}(b')$ by construction.

The basic idea behind this definition is to ensure that the neighborhood structure of the elements a' and b' is ℓ -equivalent. We use this idea to define a family of equivalence relations $(E_m^r)_{r,m \in \mathbb{N}}$. For $r, m \in \mathbb{N}$ and $\vec{a}, \vec{b} \in C^m$ let $\vec{a} E_m^r \vec{b}$ iff

- (1) $\text{suf}_{2^r}(\text{rd}(b_i)) = \text{suf}_{2^r}(\text{rd}(a_i))$ and $\text{suf}_{2^r}(\text{wrt}(a_i)) = \text{suf}_{2^r}(\text{wrt}(b_i))$ for all $1 \leq i \leq m$.
- (2) $|a_i \Delta a_j| =_{2^r} |b_i \Delta b_j|$ for all $1 \leq i, j \leq m$ and if $|a_i \Delta a_j| \leq 2^r$ then also $a_i \Delta a_j = b_i \Delta b_j$.
- (3) There is a partition X_1, \dots, X_k of $\{1, \dots, m\}$ such that for $X \neq X' \in \{X_1, \dots, X_k\}$ it holds that:
 - (a) If $i \in X, j \in X'$ it holds that $|a_i \Delta a_j| > 2^r$ (and therefore $|b_i \Delta b_j| > 2^r$).
 - (b) Let $i = \min X$. Then for all $j \in X$ it holds that $|a_i \Delta a_j| \leq \sum_{s=r+m-i}^r 2^s$ (and therefore also $|a_i \Delta a_j| \leq \sum_{s=r+m-i}^r 2^s$).
 - (c) Let $i = \min X$ and let U be the $3 \cdot 2^{r+m-i+1}$ -skeleton of $\text{cpr}_{2r+m-i+2}(a_i)$ and V be the $3 \cdot 2^{r+m-i+1}$ -skeleton $\text{cpr}_{2r+m-i+2}(b_i)$. Then for all $j \in X$ we have that a_j is compatible with U and b_j is compatible with V . Further if p_1, \dots, p_n are the positions in U that are associated with $\{a_j \mid j \in X_i\}$ and q_1, \dots, q_n are the positions in V that are associated with $\{b_j \mid j \in X_i\}$ then $(V, p_1, \dots, p_n) \equiv_{r+1} (U, q_1, \dots, q_n)$.

Lemma 4.4. For all $m \in \mathbb{N}_{>0}$ and all $\vec{a}, \vec{b} \in C^m$: If $\vec{a} E_m^0 \vec{b}$ then the mapping $a_i \mapsto b_i$ is a partial isomorphism.

Proof. We need to show that $(a_i, a_j) \in E_x \Rightarrow (b_i, b_j) \in E_x$ for all $i, j \leq m$ and all $x \in A$. Let $\vec{a}, \vec{b} \in C^m$ with $\vec{a} E_m^0 \vec{b}$. Suppose $(a_i, a_j) \in E_x$ for some $x \in A$. Then $|a_i \Delta a_j| = 1$. Hence $a_i \Delta a_j = b_i \Delta b_j$ by (2). Since the distance between a_i and a_j and between b_i and b_j is 1, there are 2^ℓ -skeletons (for some $\ell \geq m - \min\{i, j\} + 2$) U, V such that a_i and a_j can be translated into positions p, p' in U and b_i and b_j can be translated into position q, q' in V such that $(U, p, p') \equiv_1 (V, q, q')$. There are two possible types of configurations for a_i and a_j such that they can be connected by an edge. First, it might be the case that $\text{rd}(a_i) = \text{rd}(a_j)$, $\text{wrt}(a_i)x = \text{wrt}(a_j)$, and $\mu(a_i) = \mu(a_j)$. In this case $p = p'$ and therefore $q = q'$, which implies that $\text{rd}(b_i) = \text{rd}(b_j)$, $\text{wrt}(b_i)x = \text{wrt}(b_j)$, and $\mu(b_i) = \mu(b_j)$. Therefore $(b_i, b_j) \in E_x$. Second it might be that $\text{rd}(a_i)x = \text{rd}(a_j)$, $\text{wrt}(a_i) = \text{wrt}(a_j)$, and $\mu(a_j)x^{-1}$ is the largest suffix w of $\mu(a_i)$ such that wx is a prefix of $\text{wrt}(a_i)$. This property can be translated into a formula φ on (U, p, p') of quantifier rank 1. As $(U, p, p') \equiv_1 (V, q, q')$, $(V, q, q') \models \varphi$ and therefore $(b_i, b_j) \in E_x$. \blacktriangleleft

In order to prove the main technical lemma we need to construct a “small” r -equivalent words from a given word w . This is routine since it can be achieved by a simple automata-theoretic approach.

► **Lemma 4.5.** *From a given an alphabet Σ , a word $v \in \Sigma^*$, and $r \in \mathbb{N}$ one can compute an automaton \mathcal{A} in time $\exp_{r+1}(2, p(r))$ with $L(\mathcal{A}) = \{w \in \Sigma^* \mid w \equiv_r v\}$.*

Proof sketch. Construct a first-order formula φ that characterizes the r -type of v . From φ compute an automaton \mathcal{A}_φ with $L(\mathcal{A}_\varphi) = \{w \in \Sigma^* \mid w \equiv_r v\}$. One easily show via induction on r that the size of the automaton \mathcal{A} is at most $\exp_{r+1}(2, p(r))$ for some suitable polynomial p . ◀

► **Lemma 4.6.** *For all $m, r \in \mathbb{N}$ and all $\vec{a}, \vec{b} \in C^m$:*

$$\vec{a}E_m^{r+1}\vec{b} \Rightarrow \forall a \in C \exists b \in \mathcal{N}_{\exp_{r+2}(2, p(r))}(\vec{b}) : (\vec{a}, a)E_{m+1}^r(\vec{b}, b)$$

for some polynomial p .

Proof. Let $\vec{a}, \vec{b} \in C^m$ with $(\vec{a}, \vec{b}) \in E_m^{r+1}$ and let X_1, \dots, X_k be a partition of $\{1, \dots, m\}$ with the properties described in (3). Further let $X_i(\vec{a}) = \{a_j \mid j \in X_i\}$ and $X_i(\vec{b}) = \{b_j \mid j \in X_i\}$. Consider $a \in C$. We distinguish three cases. If a has distance $\leq 4 \exp_{r+2}(2, p(r))$ from ε then we choose $b = a$.

From now on suppose a has distance $> 4 \exp_{r+2}(2, p(r))$ from ε . We consider the case that a has distance $> 2^r$ from every a_i . Since the distance from ε is exactly $|\lambda(a)| + 2|\mu(a)| + |\varrho(a)|$ it follows that $|\lambda(a)| > \exp_{r+2}(2, p(a))$ or $|\mu(a)| > \exp_{r+2}(2, p(a))$ or $|\varrho(a)| > \exp_{r+2}(2, p(a))$. Basically, we want to use the $3 \cdot 2^{r+1}$ -skeleton of a to construct a suitable answer b . However, we need to cut the last 2^{r+1} read actions in order to avoid certain problems that would occur if we want to translate elements in close proximity to a into positions of the $3 \cdot 2^{r+1}$ -skeleton. Let $a' = \text{cpr}_{2^{r+2}}(a)$. Consider the $3 \cdot 2^{r+1}$ -skeleton $V = \mathcal{S}_{3 \cdot 2^{r+1}}(a')$. By Lemma 4.5 we can construct a $3 \cdot 2^{r+1}$ -skeleton W of length at most $\exp_{r+1}(2, p(r))$. From W we construct the canonical 2^{r+2} -instantiation w . Using Lemma 2.1 we can choose words u, v of length at most 2^{r+1} such that

- $\text{suf}_{2^r}(uw) = \text{suf}_{2^r}(\text{rd}(a) \text{suf}_{2^r}(\text{rd}(a))^{-1})$,
- $\text{suf}_{2^r}(wv) = \text{suf}_{2^r}(\text{wrt}(a))$,
- $\text{pref}_{2^r}(wv) = \text{pref}_{2^r}(\text{wrt}(a))$, and
- $uw \sqcap wv = w$.

Let (v_0, v_1, \dots, v_m) be the complete canon-decomposition of $\text{wrt}(a') \sqcap \text{rd}(a')$ and let (w_0, w_1, \dots, w_n) be the complete canon-decomposition of w . Let i be the index of $\mu(a')$ in (v_0, v_1, \dots, v_m) . Because $\mathcal{S}_{3 \cdot 2^{r+1}}(a') \equiv_{r+1} W$ there is a $j \in \{0, \dots, n\}$ such that $(\mathcal{S}_{2^{r+2}}(a'), i) \equiv_r (W, j)$. Now let b be the element associated to j .

Finally, if a has distance $\leq 2^r$ from some a_i then let $Y \in \{X_1, \dots, X_k\}$ be such that $i \in Y$ and let $j = \min Y$. Let U be the $3 \cdot 2^{r+m-j+1}$ -skeleton of $\text{cpr}_{2^{r+m-j+2}}(a_j)$ and V be the $3 \cdot 2^{r+m-j+1}$ -skeleton of $\text{cpr}_{2^{r+m-j+2}}(b_j)$. Then a is compatible with U . Let p_1, \dots, p_n be the positions in U that are associated with the elements $\{a_s \mid s \in Y\}$, p_{n+1} the position in U that is associated with a , and q_1, \dots, q_n be the positions associated with $\{a_s \mid s \in Y\}$ in V . Since $(U, p_1, \dots, p_n) \equiv_{r+2} (V, q_1, \dots, q_n)$ by Property (3c) there exists a q_{n+1} with $(U, p_1, \dots, p_{n+1}) \equiv_{r+1} (V, q_1, \dots, q_{n+1})$. From q_{n+1} we compute the associated element b in the $(\sum_{s=r+m-i}^r 2^s)$ -neighborhood of b_j . The construction of b ensures that Properties (1) to (3) are fulfilled for (\vec{a}, a) and (\vec{b}, b) by adding $m+1$ to Y . Hence $(\vec{a}, a)E_m^r(\vec{b}, b)$. ◀

The Lemmata 4.4 and 4.6 ensure that E_m^r -equivalent tuples are also r -equivalent.

464 ▶ **Corollary 4.7.** For all $\vec{a} \in C^m$, $a \in C$, and $r \in \mathbb{N}$ there exists an element $b \in \mathcal{N}_{\exp_{r+2}(2, p(r))}(\vec{a})$
 465 with $(\mathfrak{C}, \vec{a}, a) \equiv_r (\mathfrak{C}, \vec{a}, b)$.

466 ▶ **Lemma 4.8.** For every $a \in C$ and every r there are at most $|A|^{4r}(\min\{|\text{rd}(a)|, |\text{wrt}(a)|\} + r)$
 467 many elements in the r -neighborhood of a node $a \in C$.

468 **Proof.** every element b in the r -neighborhood of a can be characterized by the tuple $a\Delta b =$
 469 $(u, v, w, x) \in (A^{\leq r})^4$ and $\mu(b)$. Once we have fixed $a\Delta b \in (A^{\leq r})^4$ (and therefore fixed $\text{rd}(b)$
 470 and $\text{wrt}(b)$) there are at most $\min\{|\text{rd}(b)|, |\text{wrt}(b)|\} \leq \min\{|\text{rd}(a)|, |\text{wrt}(a)|\} + r$ possible values
 471 for $\mu(b)$. ◀

472 ▶ **Theorem 4.9.** $\text{FOTh}(\mathfrak{C})$ is primitive recursive.

473 **Proof.** We use the standard model-checking algorithm for first-order logic but restrict quan-
 474 tification to the $\exp_{r+1}(2, q(r))$ -neighborhood of the current variable assignment. The cor-
 475 rectness of this procedure is guaranteed by Corollary 4.7. We see that the values $|\text{rd}(a)|$
 476 and $|\text{wrt}(a)|$ are bounded by $r \exp_{r+1}(2, q(r))$. Hence, by Lemma 4.8 the algorithm needs to
 477 consider at most $|A|^{4r}(\exp_{r+1}(2, q(r)) + 1)$ many Elements, which leads to a runtime of
 478 $|\varphi| \cdot (|A|^{4r}(\exp_{r+1}(2, q(r)) + 1))^r$, which is obviously a primitive recursive function. ◀

479 5 Conclusion and Open Problems

480 We studied the Cayley-graph of the queue monoid and the logics of these graphs. Concretely,
 481 we have shown the decidability of the Cayley-graph's first order theory and the undecidability
 482 of the monadic second-order theory. This answers a question from Huschenbett et al. in [10].

483 In Table 1 is a comparison of our results compared to other fundamental data structures.

Data Structure	Transformation Monoid \mathcal{M}	$\text{FOTh}(\mathfrak{C}(\mathcal{M}, \Gamma))$	$\text{MSOTh}(\mathfrak{C}(\mathcal{M}, \Gamma))$
finite monoid	finite monoid	PSPACE [7]	PSPACE [7]
counter	$(\mathbb{Z}, +)$	2EXSPACE [18]	decidable [17]
stack	polycyclic monoid	2EXSPACE [18]	decidable [17]
queue	queue monoid	primitive recursive	undecidable

484 **Table 1** Comparison of the decidability of logics on Cayley-graphs of fundamental data struc-
 485 tures.

486 There are still some questions open relating to the queue monoid: in this paper we
 487 have given an primitive recursive but non-elementary upper bound on the complexity of
 488 the first-order theory of the queue monoid's Cayley-graph. So, one may ask for tight upper
 489 and lower bounds. Another open question concern the automaticity of the queue monoid.
 490 While it is neither automatic in the sense of Khnoussainov and Nerode [13] nor automatic
 491 in the sense of Thurston et al. [4] due to [10], we still do not know whether the Cayley-graph
 492 of the queue monoid is automatic. Finally, the decidability of the first-order theory of the
 (partially) lossy queue monoid's (cf. [14, 15]) Cayley-graph is left open as well and is worth
 to be studied.

493 References

- 494 1 Parosh A. Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. *In-*
 495 *formation and Computation*, 127(2):91–101, 1996. doi:10.1006/inco.1996.0053.

- 496 **2** Benedikt Bollig. *Formal Models of Communicating Systems: Languages, Automata, and*
497 *Monadic Second-Order Logic*. Springer-Verlag, Berlin Heidelberg, 2006. doi:10.1007/
498 3-540-32923-4.
- 499 **3** Daniel Brand and Pitro Zafiropulo. On Communicating Finite-State Machines. *Journal of*
500 *the ACM*, 30(2), 1983. doi:10.1145/322374.322380.
- 501 **4** James W Cannon, David BA Epstein, Derek F Holt, Silvio VF Levy, Michael S Paterson,
502 and William P Thurston. Word processing in groups. *Jones and Barlett Publ., Boston,*
503 *MA*, 1992.
- 504 **5** Gérard Cécé, Alain Finkel, and S. Purushotaman Iyer. Unreliable channels are easier to
505 verify than perfect channels. *Information and Computation*, 124(1):20–31, 1996. doi:
506 10.1006/inco.1996.0003.
- 507 **6** Jeanne Ferrante and Charles W. Rackoff. *The Computational Complexity of Logical The-*
508 *ories*. Number 718 in *Lecture Notes in Mathematics*. Springer, 1979. doi:10.1007/
509 BFb0062837.
- 510 **7** Erich Grädel. Finite Model Theory and Descriptive Complexity. In *Finite Model Theory*
511 *and Its Applications*, Texts in Theoretical Computer Science an EATCS Series, pages 125–
512 230. Springer, Berlin, Heidelberg, 2007. doi:10.1007/3-540-68804-8_3.
- 513 **8** James A. Green. On the structure of semigroups. *Annals of Mathematics*, pages 163–172,
514 1951.
- 515 **9** Christoph Haase, Sylvain Schmitz, and Philippe Schnoebelen. The power of priority channel
516 systems. *Logical Methods in Computer Science*, 10(4:4), 2014.
- 517 **10** Martin Huschenbett, Dietrich Kuske, and Georg Zetsche. The monoid of queue actions.
518 *Semigroup forum*, 95(3):475–508, 2017. doi:10.1007/s00233-016-9835-4.
- 519 **11** Mark Kambites. Formal languages and groups as memory. *Communications in Algebra*,
520 37(1):193–208, 2009. doi:10.1080/00927870802243580.
- 521 **12** Olga Kharlampovich, Bakhadyr Khoussainov, and Alexei Miasnikov. From automatic
522 structures to automatic groups. *Groups, Geometry, and Dynamics*, 8(1):157–198, 2014.
523 doi:10.4171/GGD/221.
- 524 **13** Bakhadyr Khoussainov and Anil Nerode. Automatic presentations of structures. In *Logic*
525 *and Computational Complexity*, volume 960 of *Lecture Notes in Computer Science*, pages
526 367–392. Springer, 1995. doi:10.1007/3-540-60178-3_93.
- 527 **14** Chris Köcher. Rational, Recognizable, and Aperiodic Sets in the Partially Lossy Queue
528 Monoid. In *STACS’18*, volume 96 of *LIPIcs*, pages 45:1–45:14. Dagstuhl Publishing, 2018.
529 doi:10.4230/LIPIcs.STACS.2018.45.
- 530 **15** Chris Köcher, Dietrich Kuske, and Olena Prianychnykova. The Inclusion Structure of Par-
531 tially Lossy Queue Monoids and their Trace Submonoids. *RAIRO - Theoretical Informatics*
532 *and Applications*, 2018.
- 533 **16** Dietrich Kuske and Markus Lohrey. Logical aspects of Cayley-graphs: The group case.
534 *Annals of Pure and Applied Logic*, 131(1):263–286, 2005. doi:10.1016/j.apal.2004.06.
535 002.
- 536 **17** Dietrich Kuske and Markus Lohrey. Logical aspects of Cayley-graphs: The monoid case.
537 *International Journal of Algebra and Computation*, 16(02):307–340, 2006. doi:10.1142/
538 S0218196706003001.
- 539 **18** Dietrich Kuske and Markus Lohrey. Automatic structures of bounded degree revisited.
540 *The Journal of Symbolic Logic*, 76(4):1352–1380, December 2011. doi:10.2178/jsl/
541 1318338854.
- 542 **19** Benoît Masson and Philippe Schnoebelen. On verifying fair lossy channel systems. In
543 *MFCS’02*, volume 2420 of *Lecture Notes in Computer Science*, pages 543–555. Springer,
544 2002. doi:10.1007/3-540-45687-2_45.

- 545 20 David E. Muller and Paul E. Schupp. The theory of ends, pushdown automata, and
546 second-order logic. *Theoretical Computer Science*, 37:51–75, January 1985. doi:10.1016/
547 0304-3975(85)90087-8.
- 548 21 Paliath Narendran and Friedrich Otto. Some results on equational unification. In *10th*
549 *International Conference on Automated Deduction*, Lecture Notes in Computer Science,
550 pages 276–291. Springer, Berlin, Heidelberg, July 1990. doi:10.1007/3-540-52885-7_94.
- 551 22 Jean-Éric Pin. Mathematical foundations of automata theory. *Lecture notes LIAFA, Uni-*
552 *versité Paris*, 7, 2010.
- 553 23 Neil Robertson and Paul D. Seymour. Graph minors, Part III: Planar tree-width.
554 *Journal of Combinatorial Theory, Series B*, 36(1):49–64, February 1984. doi:10.1016/
555 0095-8956(84)90013-3.
- 556 24 Jacques Sakarovitch. Kleene’s theorem revisited. In *Trends, Techniques, and Problems in*
557 *Theoretical Computer Science*, Lecture Notes in Computer Science, pages 39–50. Springer,
558 Berlin, Heidelberg, October 1986. doi:10.1007/3540185356_29.
- 559 25 D. Seese. The structure of the models of decidable monadic theories of graphs. *Annals of*
560 *Pure and Applied Logic*, 53(2):169–195, July 1991. doi:10.1016/0168-0072(91)90054-P.